МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Національний технічний університет України

“Київський політехнічний інститут імені Ігоря Сікорського”

«Інформатика та обчислювальна техніка» – ІОТ-2018

Матеріали наукової конференції студентів, магістрантів та аспірантів

23 – 24 квітня 2018 року

(кафедра «Автоматизованих систем обробки інформації і управління»)

Київ 2018

Зміст

УДК 519.854.2

БАБИЧ С.О.,
ГУЛЯНИЦЬКИЙ Л.Ф.

АЛГОРИТМИ РОЙОВОГО ІНТЕЛЕКТУ ДЛЯ ЗАДАЧІ ЕФЕКТИВНОГО РОЗПОДІЛУ НАВАНТАЖЕННЯ МІЖ ЕЛЕКТРОСТАНЦІЯМИ

Розглянуто підходи до розв'язування задачі динамічного розподілу навантаження для замкнутої енергетичної системи, що складається із електростанцій та споживачів. Особливістю задачі є зміна попиту на електроенергію протягом дня. Реалізовано три алгоритми ройового інтелекту, а саме оптимізація роєм частинок, алгоритм вовчої зграї та модифікація алгоритму вовчої зграї. Проведено ряд експериментів із використанням розроблених алгоритмів та аналіз отриманих результатів.

Approaches to solving the dynamic load dispatch problem for an isolated power system consisting of power stations and consumers are considered. The distinction of this task is moving during day power demand. Three swarm intelligence algorithms were used: particle swarm optimization algorithm, grey wolf optimizer algorithm and improved grey wolf optimization. A set of experiments are conducted on the use of developed algorithms and analysis of the obtained results is given.

1. Вступ

У перспективній конкуруючій галузі електроенергетики задачі економічного розподілу навантаження (ЕРН) та динамічного розподілу навантаження (ДРН) описують процес розподілу робочої потужності між електростанціями, враховуючи всі фізичні та експлуатаційні обмеження. В останні кілька десятиліть для розв'язування задачі ЕРН було застосовано ряд підходів, таких як математичне програмування (градієнтний метод) [2], лінійне програмування [3], динамічне програмування [4], метод невизначених множників Лагранжа [5], метод гілок та меж [6] та інших. Але, з огляду на особливості форми області значень цільової функції оптимізаційної задачі, зазвичай вони затримуються в локальних оптимумах цільової функції.

Алгоритм оптимізації роєм частинок (ОРЧ), вперше запропонований Кеннеді та Еберхартом [11], є ще одним сучасним прикладним методом оптимізації. В основу даного методу закладено соціально-психологічна поведінкова модель натовпу. За останні два десятиліття було розроблено багато модифікацій даного алгоритму, серед них гібридний алгоритм ОРЧ та хаотичний алгоритм ОРЧ [12].

В 2014 році з'явився новий метаевристичний алгоритм – алгоритм вовчої зграї (АВЗ) [13], який ґрунтується на моделюванні соціальної ієрархії та мисливської поведінки зграї сірих вовків. Його застосовують для розв'язання задачі ЕРН, при цьому він показав прийнятні результати на рівні із ОРЧ [14].

2. Постановка задачі

Ціль класичної задачі ЕРН [6] полягає у знаходженні мінімального значення робочої потужності для кожної електростанції:

$$f = \sum_{i=1}^{N} F_i(P_i) \to \min,$$

де, $F_i(P_i)$ – вартість палива, яке використовують для генерації робочої потужності P_i, N – кількість електростанцій. Як правило, експлуатаційні втрати кожного генератора при генерації конкретної вихідної потужності моделюються як:

$$F_i(P_i) = a_i + b_i P_i + c_i P_i^2,$$

де a_i, b_i, c_i – коефіцієнти витрат i-о генератора, P_i – це робоча потужність i-ї електростанції в МВт.

На відмінну від задачі ЕРН, задача ДРН враховує зміну попиту на електроенергію протягом дня [15]. Таким чином, задача з цільовою функцією, яка характеризуватиме вартість палива в у.о., виглядатиме наступним чином:

$$f = \sum_{t=i}^{T} \sum_{i=1}^{N} F_{ti}(P_{ti}) \to \min, \qquad (1)$$

де T – кількість запланованих періодів, N – кількість електростанцій, а $F_{ti}(P_{ti})$ – вартість палива, яке використовують для генерації робочої потужності P_i в момент часу t. Беручи до уваги особливості роботи клапанів генераторів, функцію витрат палива i-ї електростанції у (1)можна подати як суму квадратичної та синусоїдальної функції у такій формі:

$$F_{ti}(P_{ti}) = a_i P_{ti}^2 + b_i P_{ti} + c + \\ + \left| e_i \sin(y_i * (P_i^{\min} - P_{ti})) \right| ,$$

де a_i, b_i, c_i – коефіцієнти витрат i-ї електростанції, а e_i та y_i – константи, що показують вплив клапана i-ї електростанції, P_i – це робоча потужність i-ї електростанції в МВт $i = \overline{1, N}$.

Наведемо обмеження, що накладаються на замкнену систему електромережі.

1. Рівняння балансу:

$$\sum_{i=1}^{N} P_{ti} - P_{tD} - P_{tL} = 0,$$

де P_{tD} – загальна попит на електроенергію в момент часу t; P_{tL} – втрати робочої потужності в момент часу t. Величина P_{tL} розраховується наступним чином:

$$P_{tL} = \sum_{i=1}^{N} \sum_{j=1}^{N} P_{ti} B_{ij} P_{tj} + \sum_{i=1}^{n} B_{i0} P_{ti} + B_{00} .$$

де B_{ij} – i, j-й елемент квадратичної матриці коефіцієнтів втрат, що має розмірність $N \times N$, B_{i0} – i-й елемент вектора коефіцієнтів втрат, що має розмірність $1 \times N$, B_{00} – константа втрат.

2. Обмеження генерації робочої потужності:

$$P_i^{\min} \le P_{ti} \le P_i^{\max} ,$$

де $P_i^{\min}$, $P_i^{\max}$ – це мінімальна та максимальна межі реальних робочих потужностей для i-ї електростанції.

3. Заборонені зони.

Через дію клапанів та вібрації в підшипниках розглядають діапазони значень робочої потужності, при яких робота генератора неможлива або завдає великих збитків. Ці діапазони називають забороненими зонами експлуатації. Практично найкраща економія досягається шляхом уникнення експлуатації в таких місцях протягом всієї операції. Робочі зони i-ї електростанції описуються так:

$$P_i^{\min} \le P_{ti} \le P_{i,1}^{u}, \\ P_{i,j-1}^{l} \le P_{ti} \le P_{i,j}^{u}, \\ P_{i,k}^{l} \le P_{ti} \le P_i^{\max}, \\ j = \overline{2, k},$$

де l, u – нижня та верхня межі для j-ї робочої зони, а k – кількість робочих зон.

3. Алгоритм оптимізації роєм частинок

У алгоритмі ОРЧ координати кожної частинки являються собою можливий розв’язок, який пов'язаний із розташуванням та вектором швидкості інших частинок рою [12]. На швидкість частки впливають три компоненти: інерційна, когнітивна та соціальна. Інерційний компонент імітує інертну поведінку фізичних об’єктів, що продовжують рухатися деякий час по інерції в попередньому напрямку після рішення змінити його. Когнітивний компонент моделює пам'ять частинки щодо свого попереднього найкращого положення, а соціальна складова – вплив найкращого розв’язку, отриманого зграєю за весь процес роботи. Частинки рухаються у багатовимірному просторі пошуку до тих пір, поки вони не знайдуть оптимальний чи близький до нього розв’язок.

Модифіковану швидкість кожного агента можна розрахувати, використовуючи поточну швидкість та відстань від *Pbest* (найкращого розв’язку конкретного агента) та *Gbest* (найкращого розв’язку рою):

$$V_{ij}^{t} = wV_{ij}^{t-1} + C_1 r_1 (Pbest_{ij}^{t-1} - X_{ij}^{t-1}) + \\ + C_2 r_2 (Gbest_i^{t-1} - X_{ij}^{t-1}), \\ i = \overline{1, N_D} , \\ j = \overline{1, M} ,$$

де t – поточна ітерація, V_{ij}^{t} – швидкість j-ї частки в i-ому просторі для поточної ітерації t, X_{ij}^{t} – позиція j-ї частки в i-ому просторі для поточної ітерації t, w – вага

інерції, C_1, C_2 – коефіцієнти прискорення, $Pbest_{ij}^{t-1}$ – найкраща позиція j-ї частки в i-ому просторі до ітерації t, $Gbest_i^{t-1}$ – i-й компонент найкращого розв'язку рою до ітерації t, N_D – число змінних, M – кількість часток в рої, r_1, r_2 – випадкові величини рівномірно розподілені на проміжку $[0,1]$.

Отримана швидкість із виразу використовується для обчислення позиції агента:

$$X_{ij}^t = X_{ij}^{t-1} + V_{ij}^t .$$

На рисунку 1 представлена схема алгоритму ОРЧ для задачі розподілу навантаження.

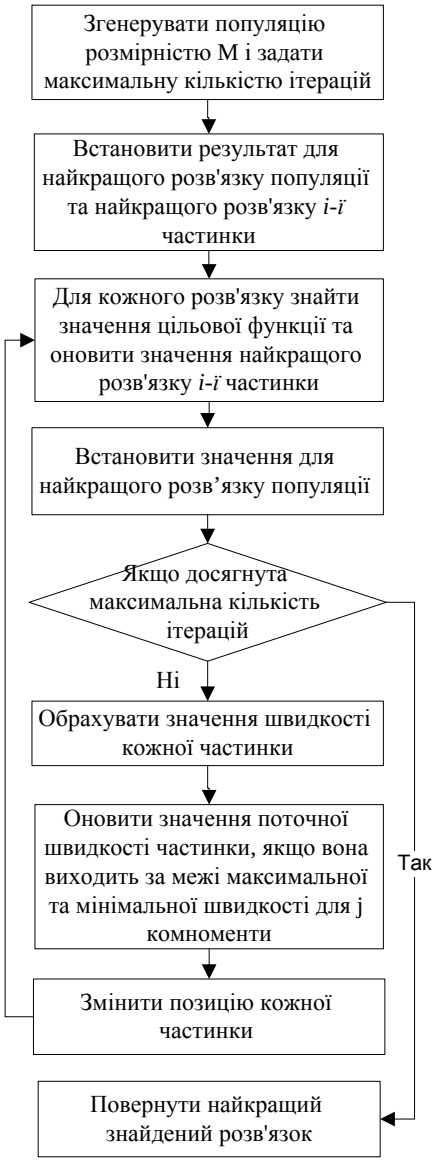


Рис.1. Схема алгоритму ОРЧ

4. Алгоритм вовчої зграї

У математичній моделі соціальної ієрархії у популяції сірих вовків альфою вважають найкращий розв'язок задачі. Відповідно, другий найкращий розв'язок називають бетою, а третій – дельтою. Розв'язки, які залишилися, приймають як омега. У алгоритмі вовчої зграї оптимізація – це моделювання процесу полювання, яким займаються альфа, бета та дельта [13]. Омега-вовки лише допомагають у пошуку здобичі.

Сірі вовки оточують здобич під час полювання. Поведінка навколишнього середовища моделюється наступним чином:

$$\overline{X}(t+1) = \overline{X_p}(t) - \overline{A}\left|\overline{C}\,\overline{X_p}(t) - \overline{X}(t)\right|,$$

$$\overline{A} = 2\overline{a}\,\overline{r_1} - \overline{a},$$

$$\overline{C} = 2\overline{r}_2,$$

де $\overline{X}$ – вектор координат розв'язку (омега вовк) на поточній ітерації t, $\overline{X_p}$ – вектор координат одного із найкращих на поточній ітерації розв'язків $p \in \{\alpha, \beta, \delta\}$, $\overline{A}$ – вектор соціальних коефіцієнтів, $\overline{C}$ – вектор когнітивних коефіцієнтів, $\overline{r_1}$ та $\overline{r_2}$ – вектори випадкових величин рівномірно розподілений на проміжку [0,1], $\overline{a}$ – вектор, компоненти якого лінійно зменшуються від 2 до 0 протягом виконання алгоритму.

На рисунку 2 представлена схема алгоритму АВЗ для задачі розподілу навантаження.

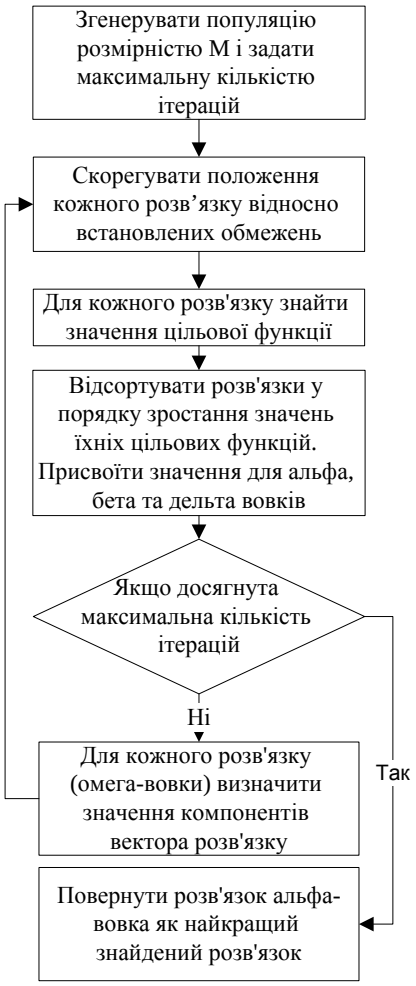


Рис.2. Схема АВЗ

5. Модифікація алгоритму вовчої зграї

Робота класичного АВЗ поділяється на дві стадії: пошуку та полювання (атаки здобичі) [13]. Перша стадія відповідає за глобальний пошук можливого оптимуму, другий же – за локальну оптимізацію. Після проведення дослідження особливостей роботи алгоритму вовчої при розв'язанні задачі ДРН, було виявлено важливу особливість – з огляду на велику кількість обмежень простір визначення цільової функції має велику кількість локальних оптимумів, через що алгоритм по завершенні першої стадії часто потрапляє в один із них і продовжує оптимізацію. Під час другого етапу отримати розв'язки, що істотно відрізнятимуться від домінуючих, неможливо, що значною мірою впливає на якість отриманого розв'язку. Для розв'язанні даної проблеми у роботі [14] запропоновано альтернативний спосіб визначення коефіцієнта соціальної складової, але цього не достатньо.

Як вже було описано, основною ідеєю вибору нових точок є поєднання впливу випадкової величини та існуючих трьох розв'язків, які із ростом кількості пройдених ітерацій збільшують свій вплив. Таким чином при переході до другого етапу усі згенеровані розв'язки будуть розміщені у околі домінуючих, і можлива область появи з кожною ітерацією звужуватиметься. У якості механізму виходу із локального оптимуму можна використати досить простий механізм – частина зграї буде не наближатись (атакувати) здобич а навпаки – віддалятись від неї (охороняти). Таким чином соціальна складова буде не звужувати область генерації нового розв'язку, а навпаки – віддаляти його від центру. Таким чином збільшується імовірність отримання нового локального оптимуму навіть під час другого етапу.

Підсумовуючи вищесказане, серед омега-вовків можна виділити дві групи – «мисливці» та «охоронці». Оскільки вони показують найгірші результати, вони будуть виконувати роль сторожі і досліджувати територію навколо зграї в пошуках нової здобичі.

Для визначення компонентів вектора розв'язку, що відповідають за «охоронців» будемо використовувати наступні формули:

$$\overline{X}_1 = \overline{X}_\alpha - \overline{A}_1 \left| \overline{C}_1 \overline{X}_\alpha - \frac{1}{2}\overline{X} \right|$$

$$\overline{X}_2 = \overline{X}_\beta - \overline{A}_2 \left| \overline{C}_2 \overline{X}_\beta - \frac{1}{2}\overline{X} \right|$$

$$\overline{X}_3 = \overline{X}_\delta - \overline{A}_3 \left| \overline{C}_3 \overline{X}_\delta - \frac{1}{2}\overline{X} \right|$$

$$\overline{X}(t+1) = \frac{\overline{X}_1 + \overline{X}_2 + \overline{X}_3}{3}$$

На рисунку 3 представлена схема модифікації АВЗ для задачі розподілу навантаження.

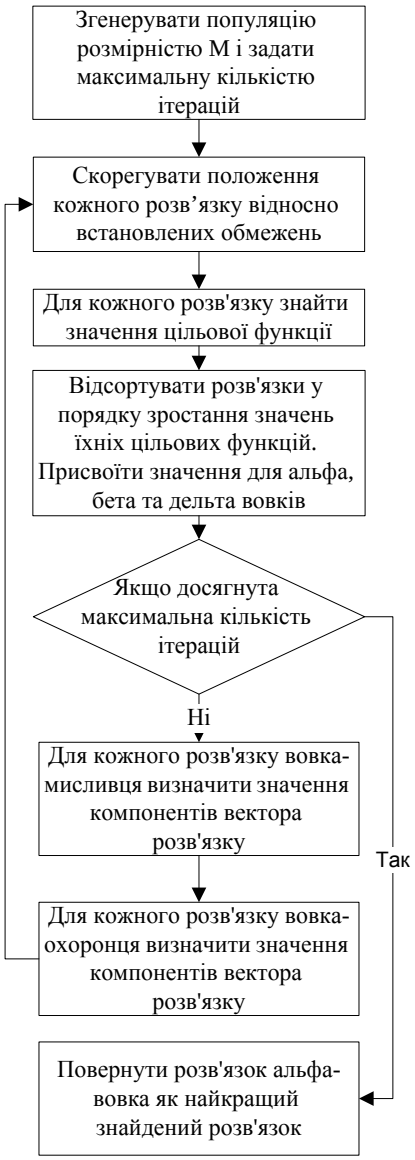


Рис.3. Схема модифікації АВЗ

6. Результати експериментів

Проведено обчислювальний експеримент із різними значеннями максимальної кількості ітерацій серед 100 тестових запусків для визначення математичного сподівання та дисперсії значень цільової функції в точці розв'язку. У якості вхідних даних було обрано задачу, наведену в [11]. Зауважимо, що для кожного тестового запуску було згенеровано нову популяцію. Результати наведені у таблицях 1 та 2.

Як можна побачити, МАВЗ та АВЗ отримали в середньому на 25% кращі за точністю результати. хоча і з більшою дисперсією значень цільової функції. МАВЗ відрізняється своєю стійкістю, про що свідчить менше значення дисперсії.

Табл. 1. Порівняння математичного сподівання для значень цільової функцій

Кількість ітерацій	ОРЧ	АВЗ	МАВЗ
100	38578,86	31767,35	30570,71
250	38413,57	30663,11	30212,5
500	38158,18	30061,90	29987,81
1000	37901,38	29757,96	29654,34
2000	38445,99	29607,20	29598,12

Табл. 2. Порівняння дисперсії для значень цільової функцій

Кількість ітерацій	ОРЧ	АВЗ	МАВЗ
100	1078,75	313,11	202,5
250	829,69	1172,15	404,91
500	1379,51	1713,92	639,38
1000	1305,37	2013,4	654,9
2000	1164,52	2164,07	714,35

7. Висновки

З огляду на отримані результати експериментів можна зробити висновок, що алгоритми ройового інтелекту можуть бути застосовані для розв'язування практичних задач ДРН. Але при цьому, з огляду на свою природу, отриманий розв'язок задачі може не повністю задовольняти обмеження задачі, через що з'являється необхідність у додатковому кроці «доопрацювання» по завершенні розв'язання.

Модифікація АВЗ продемонструвала знаходження більш точних значень цільової функції задачі у порівнянні з класичною реалізацією та алгоритмом ОРЧ.

Список літератури

1. Sinha N. Evolutionary Programming Techniques for Economic Load Dispatch / N. Sinha, R. Chakrabarti, P. K. Chattopadhyay // IEEE Transactions On Evolutionary Computation. – 2003. – V. 7. – P. 83-94.
2. Dodu J.C. An optimal formulation and solution of short-range operating problems for a power system with flow constraints / J.C.Dodu, P. Martin, A. Merlin// IEEE Proc. – 1972. – V. 60(1). – P. 54–63.
3. Parikh J. A multi-area linear programming approach for analysis of economic operation of the Indian power system / J. Parikh, D. Chattopadhyay // IEEE Trans Power Syst. – 1996. –V. 11. –P. 2–8.
4. Liang ZX. A zoom feature for a dynamic programming solution to economic dispatch including transmission losses /Z.X. Liang,J.D. Glover //IEEE Trans Power Syst. – 1992. – V. 7. – P. 50.
5. Nanda J. Economic emission dispatch with line flow constraints using a classical technique / J. Nanda, L. Hari, M.L. Kothari // IEE ProcGener Trans Distrib. – 1994. – V. 141. – P. 10.
6. Chen C.L. Branch and bound scheduling for thermal generating units / C.L. Chen, S.C. Wang // IEEE Trans Energy Convers. – 1993. – V. 184. – P. 9.
7. Perez-Guerrero R.E. Economic power dispatch with non-smooth cost functions using differential evolution / R.E. Perez-Guerrero, R.J. Cedenio-Maldonado // Proceedings of the 37th annual North American power symposium. – 2005. – P. 90.
8. Chiang C.L. Genetic-based algorithm for power economic load dispatch / C.L. Chiang // IEE ProcGener Trans Distrib. – 2007. –V. 1. – P. 9.
9. Bakirtzis A. A genetic algorithm solution to the economic dispatch problem / A. Bakirtzis, V. Petridis, S. Kazarlis // Generation, Transmission and Distribution. – 1994. – V. 141. – P. 377–382.
10. Pradhan Moumita. Oppositional based grey wolf optimization algorithm for economic dispatch problem of power system / Moumita Pradhan, Provas Kumar Roy, Tandra Pal // Ain Shams Engineering Journal. – 2017. – № 2. – P. 1-11.
11. Kennedy J. Particle Swarm Optimization / J. Kennedy // Proceedings of IEEE International Conference on Neural Networks. – 1995. – P. 1942-1948.
12. Coelho L.S. Solving economic load dispatch problems in power systems using chaotic and Gaussian particle swarm optimization approaches / L.S. Coelho, C. Lee // International Journal of Electrical Power. – 2008. – № 5 – P. 297-307.
13. Mirjalili S. Grey Wolf Optimizer / S. Mirjalili, S. M. Mirjalili, A. Lewis // Advances in Engineering Software. – 2014. – V. 69. – P. 46-61.
14. Sharma Sudhir. Economic Load Dispatch Using Grey Wolf Optimization / Sudhir Sharma, Shivani Mehta, Nitish Chopra // Journal of Engineering Research and Applications. – 2015. – № 4. – P. 128-132
15. Balamurugan R. An Improved Differential Evolution Based Dynamic Economic Dispatch with Nonsmooth Fuel Cost Function / R. Balamurugan, S. Subramanian // Electrical Systems. – 2007. – № 3.– P. 151-161.

УДК 004.432.4

Василенко В.Г.,
Ширій В.В.,
Баклан І.В.

АНАЛІЗ АБСТРАКТНИХ ТИПІВ ДАНИХ У ЙМОВІРНІСНИХ МОВАХ ПРОГРАМУВАННЯ

Сьогодні існує безліч різних імовірнісних мов програмування, які певною мірою використовують поняття теорії ймовірностей для розрахунків. Але ми хотіли б знати, які типи даних існують для вирішення ймовірнісних завдань. У даній роботі ми представляємо аналіз абстрактних типів даних на вибраних мовах імовірнісного програмування. Ми виявили кілька моментів. Перший момент полягає в тому, що всі вибрані нами мови, використовують типи даних своїх "батьківських" мов програмування. Другий полягає в тому, що для використання дистрибутивів та випадкових величин використовуються вбудовані функції або методи кожної з мов. І написання власного може призвести до деяких труднощів. Ми спробували зобразити основні сфери використання сучасних ймовірнісних абстрактних типів даних у ймовірнісних мовах програмування.

Today, there are quite a few different probabilistic programming languages that to some extent use the concepts of probability theory for their calculations. But we wanted to know what data types exist for solving probabilistic tasks. In the present paper we present an analysis of abstract data types in selected languages of probabilistic programming. We revealed several moments. The first is that all the languages we choose use the data types of their "parent" programming languages. The second is that for the use of distributions and random variables, the built-in functions or methods in each of the languages are used. And writing your own can cause some difficulties. We tried to depict the main areas of use of modern probabilistic abstract data types in probabilistic programming languages.

Ключові слова: ЙМОВІРНІСНЕ ПРОГРАМУВАННЯ, АБСТРАКТНІ ТИПИ ДАНИХ, ПАРАДИГМА ПРОГРАМУВАННЯ, МОВИ ПРОГРАМУВАННЯ

1. Вступ

На сьогоднішній день стає поширеним застосування ймовірнісних моделей, які використовуються для створення сучасного штучного інтелекту, у прикладній статистиці чи в когнітивній науці. Це пояснюється тим, що вони пов’язані з роботою над ймовірностями та їх ймовірнісними висновками [1,2]. Однак, ймовірнісні моделі мають тенденцію до збільшення їхньої складності. Тому потрібно створювати нові інструменти для забезпечення нового комплексного підходу до ймовірнісного представлення моделей. І саме ймовірнісні мови програмування це забезпечують. Мови дозволяють створювати засоби для опису складних ймовірнісних розподілів та реалізують виконання ефективного ймовірнісного висновку для довільної комп’ютерної програми.

Ймовірнісні мови програмування, в їх прості формі, розширюють добре відомі детерміновані мови програмування з примітивними конструкціями для випадкового вибору [17]. Проте з часом, відбулося створення нових інструментів для ймовірнісного виводу та зародження нових складніших ймовірнісних моделюючих програм. Наявність великої кількості ймовірнісних мов програмування змусило прийти до думки, що існує певна парадигма програмування, так зване, ймовірнісне програмування.

2. Аналіз останніх досліджень та публікацій

Основні принци дизайну мов ймовірнісного програмування були дані в [8]. Також у тій статті описується відмінності між ймовірнісним програмуванням та ймовірнісною перевіркою моделі (Probabilistic Model Checking).

Про кожну з імовірнісних мов програмування є відповідна стаття або відповідна сторінка в Інтернеті від їх авторів. Тому ми перерахуємо ті мови, про які ми будемо говорити. А саме: Church (MIT BCS/CSAIL) [5, 13], Anglican (MIT, Oxford University and DARPA PPAML) [2-4, 15], Venture (MIT BCS/CSAIL) [9, 11, 18], Infer.Net (Microsoft Research) [12], TensorFlow [6, 13] (Google) з бібліотеками

TensorFlow Distributions (Google, Columbia University) [1] та Edward (Columbia University) [7, 8, 16].

3. Об’єкти дослідження

В кожній програмування ймовірнісного програмування за допомогою абстрактних типів реалізуються основні поняття теорії ймовірностей: ймовірнісний простір, випадкова величина, ймовірність, розподіл ймовірностей. Ці поняття, на наш погляд, обов'язково повинні бути реалізовані в мовах ймовірнісного програмування. У цій статті ми проаналізуємо реалізацію основних понять теорії ймовірностей з абстрактними типами даних на імовірнісних мовах програмування. А саме: Church, Anglican, Venture, Infer.Net, TensorFlow з бібліотеками TensorFlow Distributions and Edward.

4. Church (MIT BCS/CSAIL)

Почнемо наш аналіз з Church. Church – універсальна мова програмування для опису стохастичних породжуючих процесів. Church побудована на основі моделі лямбда-числення мови LISP, з використанням чистого LISP як його детерміністична підмножина.

Ми надамо частину опису мови з [5]: «Мова Church базується на підмножині функціональної мови Scheme, як являється діалектом від LISP». Це означає, що Church використовує ті самі абстрактні типи, що і Scheme. Особливістю Church є те, що вирази являються значеннями, в той час, як самі вирази описують породжуючі моделі.

В Church є одна особливість – всі обчислення повертаються у вигляді випадкової величини [12]. Або іншими слова: значення у виразах і процедурах; якщо $v_1 \dots v_n$ є значеннями в Church, тоді як $(v_1 \dots v_n)$ - лише одне значення Church. Наприклад, у ситуації, коли дається одна випадкова величина, розподіл ймовірності часто називають випадковою величиною.

Для вказування наборів даних, потрібно застосовувати вбудовані команди, такі як *list*, *vector* і *map*. Використовуючи вбудовані типи Scheme для відображення вірогідності, Church використовує числові тип. І це може бути як ціле число або, якщо необхідно розрахувати ймовірність, раціональне.

5. Anglican (MIT, Oxford University and DARPA PPAML)

Оскільки Anglican являє собою діалект програмної мови Clojure, тому використовує ті ж абстрактні типи даних. Ось тільки Clojure застосовує ті ж типи, що і Java, що також значить, що всі значення в Anglican є регулярними посиланнями на об’єкти Java.

Для представлення наборів, Anglican, як і також Church, використовує типи *list, vector* та *map.* Вибірковий метод повертає довільну вибірку і приблизно відповідає стандартній реалізації відбіркової контрольної точки.

Для зберігання та роботи з ймовірністями, Anglican використовує бібліотеку java.lang.BigDecimal та інші класи, які можуть бути необхідні для обчислень, тому що вони завантажені в Clojure з Java. Метод спостережень повертає ймовірність реєстрації спостережень, яке приблизно відповідає стандартній реалізації контрольної точки спостережень.

Для задання розподілу використовується макрос defdist, який слідкує за визначення окремого типу для кожного розподілу так, щоб мультиметоди Clojure (або перевантажені методи) могли бути диспетчеризовані на типи розподілу за необхідності.

Додатково, Anglican забезпечує ще й випадкові процеси, які визначають послідовності випадкових величин, які не є незалежними та однаково розподіленими. Випадкові процеси визначаються за допомогою макросу defproc і реалізують протокол anglican.runtime/random-process. Цей протокол має два методи: produce (повертає розподіл в наступній випадковій величині в послідовності) та absorb (включає значення для наступної випадкової величини і повертає оновлений випадковий процес).

6. Venture (MIT BCS/CSAIL)

Venture являє собою Lisp-подібну мову вищого порядку, доповнену двома новими абстракціями: ймовірнісні сліди виконання та стохастичні процедури.

Ймовірнісні сліди виконання (Probabilistic execution traces, PET) є об'єктом першого класу, який представляє собою послідовність випадкових виборів, які робить ймовірнісна програма. Кожне програмне обчислення , яке дає результат, відповідає випадковій величині. PET служать як єдиний нативний спосіб непостійного сховища в Venture та відбивають динамічні "адреси", присвоєні протягом виконання програми явним значенням, які прийняла програма в тих адресах.

Стохастичні процедури (Stochastic procedures, SP) використовуються для інкапсуляції простих розподілів імовірності, а також простір користувача програми VentureScript і зовнішні ймовірнісні об'єкти. SP складається з пов'язаної збірки програм і мета-програм, які в сукупності описують аспекти ймовірнісної програми, важливі для його використання в моделюванні та висновках. SP розроблені, щоб дозволити простим розподілам ймовірностей, простору користувача VentureScript і зовнішнім ймовірнісним програмам розглядатися як стандартні блоки складних ймовірнісних обчислень.

Автори стверджують, що Venture використовує звичайні скалярні та символьні типи даних з мови програмування Scheme. Також у Venture існує підтримка колекцій та додаткових типів даних, що відповідають примітивному об'єкту з теорії ймовірностей та статистики. Існує підтримка стохастичної типу даних для використання в складних процедурах.

Наведемо список найбільш важливих значень: атоми (дискретні елементи без внутрішньої структури або впорядкування), числа (типи даних як ціле число, раціональне, реальне, і складне), набори (вектори і карти/map), SP.

7. Infer.NET (Microsoft Research)

Платформа Infer.NET служить для виконання Байєсового висновку в графічних моделях. Infer.NET забезпечує сучасними алгоритмами передачі повідомлень (message-passing) і статистичні процедури, необхідні для виконання висновків для широкого кола задач.

В Infer.NET можна створити три типи змінних: випадкові (значення невідомі, а апостеріорні розподіли можуть бути розраховані під час висновку), постійні (фіксовані значення), спостережувані (значення не зазначені при побудові моделі, однак дані перед виконанням висновку).

Infer.NET також дозволяє використовувати прості типи даних для створення складніших типів, на відміну від «примітивних» bool, double, int, enum, string, char. Vector та PositiveDefiniteMatrix використовуються як векторні та матричні типи для створення ймовірнісних множин. Крім того, всі вони, а також TDomain [11], ISparseList <>, IList <> можуть бути використані для дискретних, безперервних, багатоваріантних та послідовних розподілів.

Для більшої зручності та можливої простоти розробники надали методи для створення випадкових величин з різними коефіцієнтами розподілу. Вони можуть передавати випадкові величини як аргументи, наприклад, змінна <bool> замість int. У [12] ви можете побачити приклади такого використання, а також опис та синтаксис на Infer.NET. Вбудована функціональність дозволяє використовувати різні типи параметрів даних. Наприклад, з дискретним розподілом.

8. TensorFlow (Google), бібліотеки TensorFlow Distributions (Google, Columbia University) та Edward (Columbia University)

TensorFlow базується на використанні так званих тензорів. Ми дамо невелике визначення щодо тензорів. Тензори - це прості математичні об'єкти, які можуть бути використані для опису фізичних властивостей, як скаляри та вектори. Фактично тензори є лише узагальненням скалярів та векторів; скаляр - це тензор нульового рангу, а вектор - тензор першого рангу [19].

Розряд (або порядок) тензора визначається кількістю напрямків (і отже розмірністю масиву) необхідний для його опису. Наприклад, властивості, що

вимагають одного напрямку (перший ранг), можуть бути повністю описані вектором стовпця 3×1, а властивості, які потребують двох напрямків (тензори другого рангу), можуть бути описані 9 цифрами як матриця 3×3. Як такий, загалом тензор n-го рангу можна описати коефіцієнтами 3^n.

Тензори використовуються для представлення структури даних у програмах, написаних у TensorFlow. Використовуючи тензори, TensorFlow представляє ймовірнісний простір як N-розмірний масив або список. Тензор має статичний тип і динамічний розмір.

TensorFlow надає кілька можливостей для створення так званих випадкових тензорів з різними розподілами. У цьому випадку після кожного виклику та обчислення створюються нові випадкові значення.

Тензори можуть мати такі типи даних: bool, half, float, float64, uint8, int8, int16, int32, int64, complex64, complex128, string. Але ви також можете використовувати стандартні типи даних з мови Python. Наприклад, як bool, str, list або tuple.

Для TensorFlow вийшла бібліотека адаптації бачення теорії ймовірностей до сучасної глибокої навчальної парадигми кінцевих диференційованих обчислень. Вона називається TensorFlow Distributions і побудована на основі двох абстракцій: Розподілу (Distribution) та Бієктора (Bijector). Перший надає колекцію приблизно 60 дистрибутивів із швидкими, чисельно стабільними методами для вибірки, щільності логування та багатьох статистичних даних. Другий дозволяє застосовувати композиційні перетворення об'ємного масштабування (volumetracking transformation) з автоматичним кешуванням. Разом вони дозволяють створити модульну конструкцію високорозмірних розподілів та перетворень, які неможливі з попередніми бібліотеками.

Також був представлений Edward [7] – бібліотека глибокого ймовірнісного програмування, яка розширює глибоке вивчення досліджень шляхом створення нових експериментальних форм, більш швидких ітераційних циклів та кращої відтворюваності. Edward забезпечує мову випадкових величин для побудови широкого класу моделей: спрямованих графічних моделей, стохастичних нейронних мереж та програм з стохастичним потоком керування. У Edward випадкова величина – це об'єкт, параметризований тензорами. Для Edward бібліотека TensorFlow Distributions має бекенд.

9. Основні результати та висновки

Ми напишемо короткі висновки про кожну з можливих мов, обраних нами. Church досить проста в розумінні, оскільки будь-яке обчислення за допомогою вбудованих функцій дозволяє отримувати випадкову величину. І мова використовує абстрактні типи даних з функціональної мови Scheme.

Anglican використовує типи даних з мови Clojure, що, у свою чергу, приймає їх з Java. І, як нам здається, це пряме продовження ідей, які були закладені в Church, але з використанням сучасних платформ.

Venture є ще одним представником LISP-подібної мови. Але додає два нових абстракції: ймовірнісні сліди виконання та стохастичні процедури. Перший необхідний для послідовності випадкових виборів, тоді як другий – для забезпечення простих розподілів ймовірності. Використовуються типи даних мови Scheme.

За допомогою Infer.NET можна створювати різні змінні, включаючи випадкові. У той же час використовуючи дискретні, безперервні, багатоваріантні або послідовності розподілів. Також Infer.NET може використовуватися в інших мовах .NET, наприклад, в C++/CLI, F#, IronPython та інших.

Тензори є основними математичними об'єктами для мови TensorFlow. Вони дозволяють створювати багатовимірні структури випадкових величин з різними розподілами, що необхідні для обчислень. Для задання наборів використовуються як типи, які вбудовані мовою, так і з мови Python. Були представлені дві бібліотеки, я розширюю функціональні можливості мови. У TensorFlow Distributions додано

дві абстракції, які разом забезпечують модульну побудову високорозмірних розподілів та перетворень. Інша бібліотека – Edward – дозволяє розробляти складні ймовірнісні моделі та алгоритми для них.

Ми виділили кілька основних моментів. Перший полягає в тому, що всі мови, які були обрані, використовують типи даних своїх "батьківських" мов програмування. Другий полягає в тому, що для використання розподілів та випадкових величин використовуються вбудовані функції або методи кожної з мов. І список цих розподілів може бути різним. Залежно від різних факторів (досвід розробки, знання в галузі теорії ймовірностей, тощо), розвиток власних ймовірнісних понять може викликати плутанину.

Перелік посилань

1. Alemi A., Dillon J.V., Langmore I., *Tensorflow Distributions*. 2017p.
2. Anglican Homepage. URL: https://probprog.github.io/anglican/index.html (дата звернення 28.03.2018).
3. Anglican Language syntax, URL: https://probprog.github.io/anglican/ index.html (дата звернення 28.03.2018).
4. Anglican Inference methods, URL: https://probprog.github.io/anglican/ inference/index.html (дата звернення 28. 03.2018).
5. Computation in Church, URL: http://projects.csail.mit.edu/church/wiki/ Computation_in_Church (дата звернення 25.03.2018).
6. Constants, Sequences, and Random Values page, URL: https://www.tensorflow.org/api_guides/python/constant_op (дата звернення 28.03.2018).
7. Edward Homepage, URL: http://edwardlib.org/ (дата звернення 27.11.2017).
8. Developing Custom Random Variables URL: http://edwardlib.org/api/model-development (дата звернення 27.11.2017).
9. Gordon A. D., Henzinger T. A., Nori A. V., Rajamani S. K. (2014, May). Probabilistic programming. *Proceedings of the on Future of Software Engineering*. 2014. PP. 167-181.
10. Lu A. Venture: an extensible platform for probabilistic meta-programming: дис. канд. техн. наук.
11. Mansinghka V., Selsam D., Perov Y. Venture: a higher-order probabilistic programming platform with programmable inference.
12. Infer.NET 2.6, URL: http://research.microsoft.com/infernet (дата звернення 14.03.2018).
13. Probability Theory and The Meaning of Probabilistic Programs, URL: http://projects.csail.mit.edu/church/wiki/Probability_Theory_and_The_Meaning_of_Probabilistic_Programs (дата звернення 14.03.2018).
14. TensorFlow Homepage, URL: https://www.tensorflow.org/ (дата звернення 27.03.2018).
15. Tolpin D., van de Meent J. W., Yang H., Design and Implementation of Probabilistic Programming Language Anglican. 2014.
16. Tran, D., Kucukelbir, A., Dieng, A. B., Rudolph, M., Liang, D., & Blei, D. M.: Edward: A library for probabilistic modeling, inference, and criticism.
17. Vasilenko V., Shyrii V., Baklan I. Modern programming paradigm - probabilistic programming. *In XIIV International scientific conference "Intellectual systems of decision-making and problems of computational intelligence"*. 2017.
18. Venture Homepage, URL: http://probcomp.org/venture/ (дата звернення 14.03.2018).
19. What is a Tensor. URL: https://www.doitpoms.ac.uk/tlplib/tensors/ what_is_tensor.php (дата звернення 26.02.2018).

УДК 519.854.2

ГАЛКІНА Г.А.
ГУЛЯНИЦЬКИЙ Л.Ф.

ЗАДАЧА ОПТИМАЛЬНОГО ПЛАНУВАННЯ ВИКОНАННЯ ЧАСТКОВО ВПОРЯДКОВАНИХ ЗАВДАНЬ ЗА НАЯВНОСТІ СПІЛЬНОГО ДИРЕКТИВНОГО ТЕРМІНУ ТА РІЗНОІ ПРОДУКТИВНОСТІ ВИКОНАВЦІВ

Розглядається задача планування роботи на ітерацію гнучкої методології Скрам. Вона розглядається як задача оптимального планування завдань за наявності різної продуктивності виконавців. Для цієї задачі запропоновано алгоритм її розв'язування шляхом поділу на дві підзадачі та розв'язування їх послідовно. Для першої підзадачі використано існуючий алгоритм, для другої запропоновано алгоритм детермінованого локального пошуку та алгоритм імітаційного відпалу. Наведено ключові аспекти реалізації для запропонованих алгоритмів та порівняння результатів їх роботи на 16 наборах тестових даних.

The problem of scheduling work for an iteration in Scrum methodology is considered. It is considered as an optimal scheduling problem for machines with different unrelated productivity. A heuristic algorithm to solve the considered problem is suggested, it is based on the idea of splitting the problem into two different subsequent subproblems. An existing algorithm is used to solve the first subproblem while for the second subproblem a deterministic local search and simulated annealing algorithms are suggested. Key implementation points are given for the suggested algorithms and a comparison of the suggested algorithms' results when used for 16 input data sets is performed.

1. Вступ

Планування проектів сьогодні є областю, в якій ведуться активні дослідження. Задача планування роботи на проекті на сьогоднішній день є однією з найбільш розповсюджених через її теоретичну складність та практичний зміст. Одним із різновидів підходів до планування проектів є гнучкі методології, які набувають все більшої популярності з плином часу, прикладом такої методології є Скрам. Серцем Скрам є Спринт, з часовими рамками в місяць або менше, в результаті якого створюється "завершений", цінний та потенційно готовий до випуску Інкремент продукту.[1] Робота, що її виконують під час Спринту, планується під час наради із Планування Спринту. План дій розробляється при спільній роботі цілої Скрам Команди. Для Спринту тривалістю в місяць часові рамки зустрічі становлять вісім годин. Для більш коротких Спринтів на планування виділяють менше часу, пропорційно загальній довжині Спринту. Приміром, для двотижневого Спринту планування займе не більше чотирьох годин. [1] Саме тому автоматизація процесу планування роботи для одного Спринту принесе значну економію часу команди, яка працює над проектом. В цій роботі розглянуто задачу планування Спринту як задачу оптимального планування завдань за наявності різної продуктивності виконавців. Характеристиками розглянутої задачі є наявність загального директивного строку для всіх завдань, різна непропорційна продуктивність виконавців, наявність відношення часткового строгого впорядкування між завданнями та визначена для кожного з завдань важливість.

2. Формалізація задачі

Задача полягає у максимізації цільової функції:

$$\sum_{i=1}^{m}\sum_{j=1}^{n} x_{ij} v_j, \qquad (1)$$

яка задає сумарну важливість усіх завдань, що будуть виконані, де

$$x_{ij} = \begin{cases} 1, \textit{якщо учасник } i \textit{ виконує завдання } j \\ 0, \textit{інакше} \end{cases}, i = \overline{1,m},\ j = \overline{1,n}. \qquad (2)$$

Також позначимо s_j час початку виконання завдання j, $j \in N$, причому:

$$s_j = \begin{cases} 0, \textit{якщо завдання } j \textit{ не буде виконане}, \\ > 0, \textit{якщо завдання } j \textit{ буде виконане}. \end{cases} \quad (3)$$

Обмеження задачі:

$$0 \le \sum_{i=1}^{m} x_{ij} \le 1, \quad (4)$$

$$\sum_{i=1}^{m} x_{ij} = 1,\ j \in Q. \quad (5)$$

Для всіх $j \in N, k \in N$ таких, що $x_{ij} = x_{ik}$ для всіх $i = \overline{1,m}$ та $\sum_{i=1}^{m} x_{ij} = 1, \sum_{i=1}^{m} x_{ik} = 1$:

$$s_j < s_k \to s_j + t_{ij} < s_k. \quad (6)$$

Для всіх завдань j, $j \in N$ таких, що $s_j > 0$:

$$s_j + \sum_{i=1}^{m} x_{ij} t_{ij} \le d. \quad (7)$$

Для всіх j, $j \in N \setminus Q$, для яких $\sum_{i=1}^{m} x_{ij} = 1$ та є завдання k, що їм передують:

$$s_k < s_j,\ s_k + \sum_{i=1}^{m} x_{ik} t_{ij} < s_j. \quad (8)$$

Обмеження (4) задає можливість виконання кожного з завдань лише одним учасником команди. (5) задає обов'язкове виконання всіх завдань j, $j \in Q$. (6) – неперетинання запланованих завдань для одного учасника. Обмеження (7) задає виконання всіх завдань в розкладі до настання директивного терміну, а (8) – виконання завдань відповідно до відношення часткового строгого впорядкування.

3. Декомпозиційний підхід

Запропонуємо наступний алгоритм розв'язування сформульованої задачі. Розглянемо її як сукупність двох підзадач. Спочатку необхідно спланувати виконання обов'язкових завдань (підзадача 1), а після цього на час, що залишається, запланувати виконання такого набору необов'язкових завдань, який дасть максимально можливу важливість (підзадача 2). Логічним буде в якості цільової функції для підзадачі 1 обрати мінімізацію сумарного часу, який буде витрачено на виконання обов'язкових завдань, оскільки наступним кроком є додавання необов'язкових завдань і на них також необхідний час.Тоді на початок розв'язання підзадачі 2 отримаємо деякий існуючий розподіл обов'язкових завдань на всіх членів команди. З нього випливає, що для кожного члена команди вже є декілька завдань, і це необхідно враховувати при складанні остаточного розкладу.

Математична модель підзадачі 1 (планування обов'язкових завдань)

Задача полягає у мінімізації цільової функції

$$\sum_{i=1}^{m} \sum_{j=1}^{q} x_{ij} t_{ij}, \quad (9)$$

яка задає сумарний час, витрачений на виконання всіх обов'язкових завдань, де:

$$x_{ij} = \begin{cases} 1, \textit{якщо учасник } i \textit{ виконує завдання } j, \\ 0, \textit{інакше} \end{cases}, i = \overline{1,m},\ j = \overline{1,n}. \quad (10)$$

Обмеження задачі:

$$\sum_{i=1}^{m} x_{ij} = 1, j \in Q. \quad (11)$$

Для всіх учасників $i, i \in \{1,2,..,m\}$:

$$\sum_{j=1}^{q} x_{ij} t_{ij} \le d. \quad (12)$$

Обмеження (11) задає виконання кожного з обов'язкових завдань одним учасником команди. (12) задає виконання всіх обов'язкових завдань в розкладі кожного з учасників до настання директивного терміну. Сформульована математична модель описує узагальнену задачу про призначення. Ця класична задача комбінаторної оптимізації є NP-складною [8], для неї вже було розроблено велику кількість різноманітних алгоритмів. Сформулюємо математичну модель підзадачі 2. Для неї ми матимемо додаткові вхідні дані після розв'язування підзадачі 1 – розподілені між учасниками обов'язкові завдання з Q. Позначимо їх як b_j, $j \in Q$. Очевидно, що $b_j \in \{1,2,..,m\}$.

Математична модель підзадачі 2 (планування необов'язкових завдань)

Задача полягає у максимізації цільової функції

$$\sum_{i=1}^{m}\sum_{j=1}^{n} x_{ij} v_j, \qquad (13)$$

яка задає сумарну важливість усіх завдань, що будуть виконані, де

$$x_{ij} = \begin{cases} 1, \textit{якщо учасник } i \textit{ виконує завдання } j, \\ 0, \textit{інакше} \end{cases}, i = \overline{1,m},\ j = \overline{1,n}. \qquad (14)$$

Також позначимо s_j час початку виконання завдання j, $j \in N$, причому:

$$s_j = \begin{cases} 0, \textit{якщо завдання } j \textit{ не буде виконане}, \\ > 0, \textit{якщо завдання } j \textit{ буде виконане}. \end{cases} \qquad (15)$$

Обмеження задачі:

$$0 \le \sum_{i=1}^{m} x_{ij} \le 1, \qquad (16)$$

$$x_{b_j j} = 1, j \in Q. \qquad (17)$$

Для всіх $j \in N, k \in N$ таких, що $x_{ij} = x_{ik}$ для всіх $i = \overline{1,m}$ та $\sum_{i=1}^{m} x_{ij} = 1, \sum_{i=1}^{m} x_{ik} = 1$:

$$s_j < s_k \to s_j + t_{ij} < s_k. \qquad (18)$$

Для всіх завдань j, $j \in N$ таких, що $s_j > 0$:

$$s_j + \sum_{i=1}^{m} x_{ij} t_{ij} \le d. \qquad (19)$$

Для всіх j, $j \in N \setminus Q$, для яких $\sum_{i=1}^{m} x_{ij} = 1$ та є завдання k, що їм передують:

$$s_k < s_j,\ s_k + \sum_{i=1}^{m} x_{ik} t_{ij} < s_j. \qquad (20)$$

Обмеження (16) задає можливість виконання кожного з завдань лише одним учасником команди. (17) задає виконання обов'язкових завдань саме тими учасниками, яких було визначено в результаті розв'язування підзадачі 1. (18) – неперетинання запланованих завдань для одного учасника, (19) задає виконання всіх завдань в розкладі до настання директивного терміну. Обмеження (20) задає виконання завдань відповідно до відношення часткового строгого впорядкування.

4. Алгоритми розв'язування отриманих підзадач

Для підзадачі 1 використано алгоритм динамічного програмування для розв'язування задачі про рюкзак [2] та його модифікацію для узагальненої задачі про призначення з [3]. Для підзадачі 2 використано жадібний алгоритм для знаходження початкового розв'язку та алгоритми детермінованого локального пошуку (ДЛП) та імітаційного відпалу (AIB) для покращення знайденого початкового розв'язку. Позначимо $P = N \setminus Q, |P| = p, p = n - q$.

Детермінований локальний пошук

Схема розробленого алгоритму є класичною схемою алгоритму локального пошуку з [4].

Визначення околів та генерація чергової точки околу. Використаємо кільцевий генератор точок околу, тобто такий, що продовжує свою роботу з попереднього місця замість перегляду всіх точок, які вже були переглянуті. Вважатимемо два розв'язки сусідніми, якщо вони розрізняються складом завдань на одне завдання, тобто розв'язок 1, який має набір завдань

$$K_1 = \{ i \mid i \in N \},$$

та розв'язок 2, який має набір завдань

$$K_2 = K_1 \setminus \{ r \} \cup \{ j \},\ j \in N \setminus K_1$$

є сусідніми.

Означення 1. Відстанню зсуву вниз d_{down} для деякого завдання j, $j \in N$, яке в поточному розв'язку буде виконане учасником $i, i \in [1,..,m]$, починаючи у момент часу $s_j > 0$ та за час t_{ij}, є проміжок часу, на який можна посунути ближче до закінчення директивного терміну всі завдання $k \in N \setminus \{ j \} = K$, які у розкладі учасника i будуть виконані після завдання j, тобто $x_{ik} = 1, s_k > s_j + t_{ij}$. Позначимо завдання, яке можна виконати тільки після деякого завдання $k \in K$, як $succ(k), succ(k) \in N \setminus \{k\}$. Також позначимо завдання, яке заплановано на виконання

після завдання $k \in K$, як $after_k, after_k \in N \setminus \{k\}$; завдання, яке заплановано на виконання до завдання $k \in K$, як $before_k, before_k \in N \setminus \{k\}$; завдання, яке має бути виконано до деякого завдання $k \in K$, як $prev(k), prev(k) \in N \setminus \{k\}$. Проміжок часу для кожного завдання $k \in K$ в загальному випадку обчислюється таким чином:

$$d_{down}(k) = s_{succ(k)} - (s_k + t_{ik}).$$

Тоді загальна відстань зсуву вниз обчислюється наступним чином:

$$d_{down} = \min_{k \in K}\{d_{down}(k)\} \quad (21)$$

Означення 2. Відстанню зсуву вверх d_{up} для деякого завдання $j, j \in N$, яке в поточному розв'язку буде виконано учасником $i, i \in [1,..,m]$, починаючи у момент часу $s_j > 0$ та за час t_{ij}, є проміжок часу, на який можна посунути ближче до моменту початку цього завдання всі завдання $k \in N \setminus \{j\} = K$, які у розкладі учасника i будуть виконані після завдання j, тобто $x_{ik} = 1, s_k > s_j + t_{ij}$, якщо це завдання $j, j \in N$ вилучити з поточного розкладу для учасника i. Проміжок часу для кожного завдання $k \in K$ в загальному випадку обчислюється таким чином:

$$d_{up}(k) = \min\{s_k - (s_{before_k} + t_{ibefore_k}), s_k - (s_{prev(k)} + t_{iprev(k)})\}$$

Тоді загальна відстань зсуву вверх обчислюється наступним чином:

$$d_{up} = \min_{k \in K}\{d_{up_k}\}. \quad (22)$$

На рисунку 1 наведено схему процедури 1.

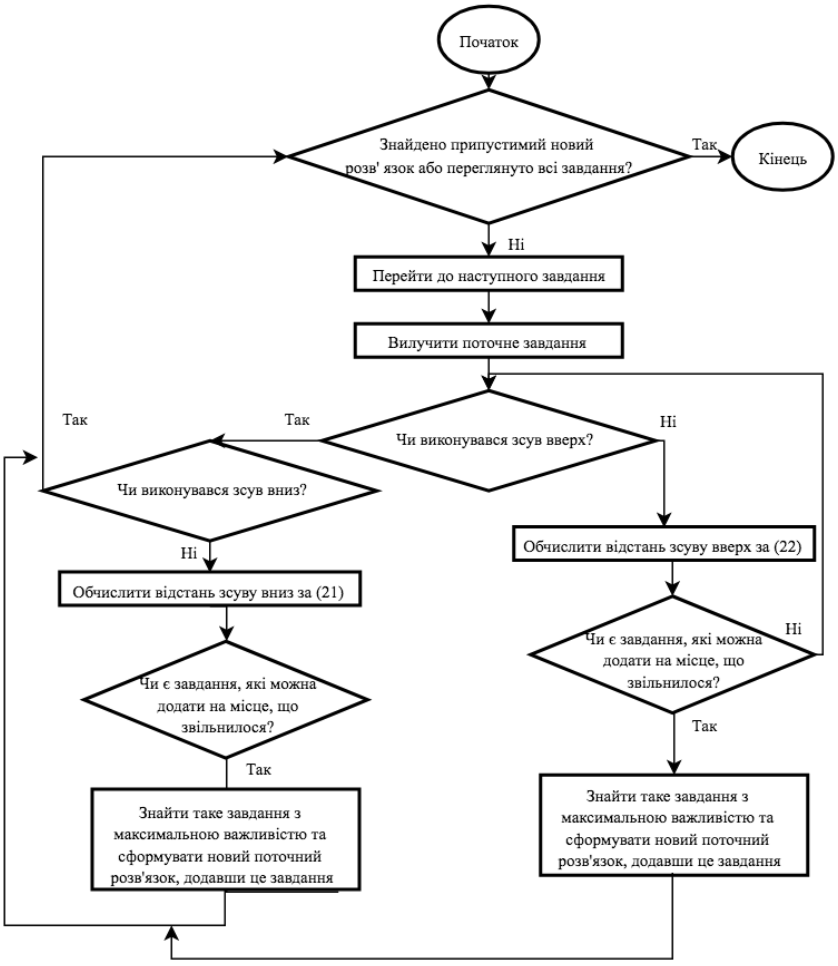


Рис.1. Схема процедури генерації сусіднього розв'язку

Критерій завершення перегляду точок у поточному околі та переходу до наступного.

Перехід до наступного розв'язку здійснюється при знаходженні першого кращого за поточний розв'язок сусіднього розв'язку.

Спосіб обчислення зміни цільової функції при переході до нового поточного варіанта

Нехай з поточного розв'язку було вилучено деяке завдання $i, i \in N$ та його було замінено завданням $j, j \in N \setminus \{i\}$. Тоді зміна цільової функції обчислюється таким чином:

$$\Delta = v_j - v_i \qquad (23)$$

Критерій завершення

В якості критерію завершення обрано відсутність кращих варіантів в околі поточного розв'язку або перевищення заданої максимальної кількості ітерацій.

Алгоритм імітаційного відпалу

Схема роботи алгоритму, що пропонується, є класичною схемою алгоритмів імітаційного відпалу з [4]. Наведемо ключові аспекти реалізації.

Поняття околу та перебір точок в околі, тобто спосіб генерації точок

Для перебору точок в околі використовується процедура 1.

Імовірність переходу від поточного варіанта до нової точки

Нехай для розв'язку S знайдено деякий припустимий сусідній розв'язок S_{new}. Тоді спочатку обчислюється різниця цільових функцій Δ за формулою (23). Після цього генерується випадкове число $r = random[0,1]$. Тоді імовірність переходу до нового розв'язку обчислюється так:

$$p = e^{\frac{-\Delta}{T}}. [4] \qquad (24)$$

Якщо $r < \min\{p,1\}$, то перехід відбувається.

Принцип рівноваги

Нехай є початковий розв'язок S_{start}, і в ньому $n_{S_{start}}$ завдань. Тоді довжина прогону визначається таким чином:

$$length = \frac{n_{S_{start}}}{m}. \qquad (25)$$

А f_i обчислюється для прогону як довжиною $length$ таким чином:

$$f_i = \frac{\sum_{w=1}^{length} f_w}{length} \qquad (26)$$

Температурний розклад (правило зміни значення параметра T)

Зміна температури відбувається від значення $T_{\max} = 1$ до значення $T_{\min} = 0.00001$. Наступне значення параметру T_r на ітерації під номером r обчислюється за формулою

$$T_r = T_{r-1} \cdot \alpha, \qquad (27)$$

де $\alpha = 0.9$.

5. Результати дослідження

Результати роботи алгоритмів наведено у таблиці 1.

Табл.1. Порівняння результатів роботи алгоритмів

№	m	n	d	start	ДЛП	ДЛП / start	АІВ	АІВ / start
1	7	116	40	1232	1924	1.56	1949	1.58
2	9	55	40	777	1155	1.48	1164	1.5
3	9	67	40	1302	2208	1.7	2281	1.75
4	6	102	40	1130	2077	1.84	2088	1.85
5	5	65	80	104	164	1.58	164	1.58
6	7	141	80	1067	1853	1.74	1864	1.75
7	9	89	80	747	1297	1.74	1304	1.75
8	10	65	80	941	1333	1.42	1342	1.43
9	7	133	120	1149	2148	1.87	2299	2
10	8	87	120	1639	2191	1.34	2212	1.35
11	8	138	120	1713	2617	1.53	2617	1.53
12	10	67	120	1793	2569	1.43	2602	1.45
13	4	91	160	454	664	1.46	665	1.47
14	5	142	160	530	1231	2.32	1335	2.52
15	6	28	160	97	127	1.31	128	1.32
16	9	60	160	713	1067	1.5	1125	1.58

Для АІВ було виконано 50 запусків з того ж самого початкового розв'язку, обрано найкращий результат з цих 50 запусків. З результатів можна зробити висновок, що обидва запропонованих алгоритми дають можливість отримати розв'язок, який є суттєво кращим за початковий. В загальному результати роботи обох алгоритмів є близькими за значеннями, але зі збільшенням m та n різниця між результатами ДЛП та АІВ повільно зростає.

6. Висновки

Наведено постановку задачі планування роботи на Спринт як задачі планування за умови наявності різної непропорційної продуктивності виконавців та часткового впорядкування завдань. Запропоновано алгоритм розв'язування поставленої задачі шляхом розбиття її на дві підзадачі. Для першої підзадачі використано існуючий алгоритм, для другої підзадачі запропоновано алгоритми детермінованого локального пошуку та імітаційного відпалу. Наведено результати розв'язування 16 екземплярів задачі обома алгоритмами. Алгоритм імітаційного відпалу в загальному випадку дозволяє отримати кращі результати, ніж алгоритм детермінованого локального пошуку.

7. Перелік посилань

1. Посібник зі Скраму [Електронний ресурс] / К. Швабер, Д. Сазерленд – Режим доступу до ресурсу: https://www.scrumguides.org/docs/scrumguide/v1/Scrum-Guide-UA.pdf

2. Lecture 13: The Knapsack Problem [Електронний ресурс] – Режим доступу до ресурсу: http://www.es.ele.tue.nl/education/5MC10/Solutions/knapsack.pdf

3. Cohen R. An efficient approximation for the Generalized Assignment Problem / R. Cohen, L. Katzir, D. Raz // Information Processing Letters. – 2006, vol. 100. – №4. – P.162-166.

4. Гуляницький Л.Ф. Прикладні методи комбінаторної оптимізації: навч. посіб. / Л. Ф. Гуляницький, О. Ю. Мулеса. – К. : Видавничо-поліграфічний центр "Київський університет", 2016. – 142 с.

УДК 004.92

ГРИГОРОВИЧ Б.А.,
ЛЯШКО С.І.

ІМІТАЦІЙНЕ МОДЕЛЮВАННЯ 6DOF-ДИНАМІКИ РОЗДІЛЮВАЧА НАМАГНІЧЕНИХ ТІЛ

Була розглянута проблема моделювання та візуалізації роботи розділювача намагнічених тіл, які рухаються в просторі і можуть обертатися. Диференціальні рівняння, які описують рух залізної руди під впливом однорідного магнітного поля і сили тяжіння, були розв'язані за допомогою чисельних методів. Була зображена траєкторія руху тіла в розподілювачі.

The problem of modeling and visualization of the work of the magnetized body separator, which moves in space and can be rotated, was considered. Differential equations describing the movement of iron ore under the influence of a homogeneous magnetic field and gravity were solved by numerical methods. The trajectory of the body motion in the distributor was depicted.

1. Вступ

Імітаційне моделювання та комп'ютерна візуалізація, як методи аналізу різноманітних наукових даних, знаходять широке застосування як в теоретичних, так і в експериментальних дослідженнях. Аналіз сучасного стану розвитку інтелектуальних інформаційних технологій дозволяє зробити висновок, що у візуалізації як новітній інтелектуально-інформаційній технології базовою компонентою і механізмом дії є принцип образної обробки інформації.

Візуалізація просторових даних використовується в основному в задачах наукової візуалізації. Наукова візуалізація – це створення графічних образів, які в максимально інформативній формі відтворюють значущі аспекти досліджуваного процесу чи явища. При цьому великий обсяг результатів моделювання подається в компактній формі, яка легко сприймається. Подання у вигляді графічних образів дозволяє досліднику побачити досліджувану систему або процес зсередини, що було б неможливим без візуалізації.

Задача моделювання та візуалізації роботи розділювача намагнічених тіл, які рухаються в просторі і можуть обертатися є важливою прикладною задачею, яка на сьогодні не вирішена.

Наведені міркування обґрунтовують актуальність обраної тематики дослідження.

Метою дослідження є побудова імітаційної моделі 6DOF-динаміки розділювача намагнічених тіл.

Для досягнення поставленої мети необхідно розв'язати такі задачі:

- виконання аналізу предметної області;
- побудова математичної моделі руху намагніченого тіла з шести ступенями вільності в зовнішніх гравітаційному та магнітному полях;
- дослідження та обрання методів та засобів реалізації;
- розроблення алгоритмів та проектування програмного забезпечення;
- реалізація розроблених алгоритмів у вигляді програмного забезпечення візуалізації динаміки розділювача намагнічених тіл з шести ступенями вільності.

Об'єкт дослідження – процеси побудови імітаційних моделей та візуалізації даних.

Предмет дослідження – методи і засоби побудови алгоритмів та програмної реалізації імітаційної моделі динаміки об'єктів з шести ступенями вільності.

2. Опис проблеми та критерії її розв'язання

На тіла, вміщені в неоднорідне магнітне поле, діє сила, що залежить від магнітної проникності тіла і ступеня неоднорідності поля. У парамагнетиків $\mu > \mu_0$ і об'ємна густина сили спрямована в бік збільшення індукції поля; у діамагнетиків $\mu < \mu_0$ і об'ємна густина сили спрямована в бік зменшення

індукції поля. Отже, поведінка магнітних і немагнітних тіл в магнітному полі відрізняється. Зокрема, магнітне і немагнітне тіла, які починають рух з однаковою швидкістю з однієї і тієї ж області поля, набудуть різних прискорень і будуть рухатися по різних траєкторіях. Фізичні процеси, що відбуваються в феромагнітних матеріалах під впливом магнітного поля, дуже складні, і побудова теоретичних моделей, в яких детально враховують всі властивості цих речовин і тіл, які з них складаються, не тільки не можлива, але і не доцільна. Тому, для обґрунтування працездатності методу і оцінки його ефективності використано досить прості ідеалізовані моделі, які, проте, відображають суттєві властивості поведінки тіл у зовнішньому магнітному полі. Математична модель складається з моделі розподілу поля і системи диференціальних рівнянь, що описують рух тіла в цьому полі. В цьому випадку рівняння руху мають такий вигляд:

$$\begin{cases} \dot{\vec{x}} = \vec{v} \\ \dot{\vec{v}} = \overline{\mu} v_s B_{s,r} \vec{e}_r - g\vec{e}_3 \\ \dot{\vec{\nu}} = \overline{\alpha}(\vec{n} \times \vec{\nu}) \\ \dot{\vec{n}} = \overline{\mu}(\vec{\nu} \times \vec{B}) \end{cases}$$

де $\vec{x}$ – вектор положення тіла в просторі,

$\vec{v}$ – швидкість тіла,

$\vec{\mu}$ – магнітна вісь тіла,

$\vec{n} = \frac{1}{m}\vec{m}$ – питомий магнітний момент,

m – маса тіла,

$\overline{\alpha} = m\alpha = \frac{1}{I/m}$ – величина, обернена моменту інерції кулі,

$\overline{\mu} = \frac{\mu}{m}$ – питомий магнітний момент,

g – прискорення вільного падіння.

При виконанні роботи було використано чисельні методи розв'язування диференціальних рівнянь, методи аналізу, синтезу, математичного моделювання, моделювання інформаційних систем, об'єктно-орієнтованого аналізу і програмування.

При програмній реалізації імітаційної моделі були використані наступні засоби:

- Python для алгоритмічної та серверної частини за стосунку;
- JavaScript для клієнтської частини застосунку;
- Бібліотека SciPy;
- середовище розробки PyCharm.

Новизна роботи: вперше розроблено алгоритми та програмне забезпечення, що описують 6DOF-динаміку намагнічених тіл в неоднорідному магнітному полі.

Практична цінність роботи: розроблене програмне забезпечення є підґрунтям для розроблення системи візуалізації 6DOF-динаміки розділювача намагнічених тіл.

3. Висновки

Побудовано алгоритми, що описують 6DOF-динаміку розділювача намагнічених тіл та виконано їх програмну реалізацію. Була розглянута проблема моделювання та візуалізації роботи розділювача намагнічених тіл, які рухаються в просторі і можуть обертатися. Диференціальні рівняння, які описують рух залізної руди під впливом однорідного магнітного поля і сили тяжіння, були розв'язані за допомогою чисельних методів. Була зображена траєкторія руху тіла в розподілювачі.

Список літератури

1. Zub S. S. Magnetic levitation in Orbitron system // Problems of atomic science and technology, 2014, N5 (93). Series: Nuclear Physics Investigations (63), p.168-176.

2. Zub S. S. Orbitron. Part I. Stable orbital motion of magnetic dipole in the eld of permanent magnets, math-ph/arXiv:1205.4203 (2012).

3. Matthew M. Peet. Spacecraft and Aircraft Dynamics. Lecture 9: 6DOF Equations of Motion [Електронний ресурс]. – Режим доступу: https://www.coursehero.com/file/9098395/ 441Lecture9/

УДК 004.946

ГУЛАК О.С.,
ОЛІЙНИК Ю.О.

ЗАСТОСУВАННЯ АЛГОРИТМІВ ОДНОЧАСНОЇ ЛОКАЛІЗАЦІЇ ТА КАРТОГРАФУВАННЯ SLAM ПРИ ФОРМУВАННІ ДОПОВНЕНОЇ РЕАЛЬНОСТІ

У даній статті розглянуто існуючі рішення щодо побудови та позиціонування елементів доповненої реальності (Augmented Reality, AR) на мобільних пристроях. Проведено аналіз наявних проблем при вирішенні задачі позиціонування віртуальних об'єктів. Розглянуто алгоритми одночасної локалізації та картографування SLAM. Наведено практичні результати інтеграції алгоритмів SLAM для кращого визначення позицій віртуальних об'єктів та орієнтації мобільного пристрою у навколишньому середовищі.

This article examines existing solutions for building and positioning Augmented Reality (AR) elements on mobile devices. Analysed the existing problems in positioning of virtual objects. The algorithms of simultaneous localization and mapping SLAM are considered. The practical results of integration of SLAM algorithms are presented for better definition of positions of virtual objects and orientation of mobile device in the environment.

1. Вступ

У даній статті розглянуто існуючі рішення щодо побудови та позиціонування елементів доповненої реальності на мобільних пристроях. Проведено аналіз наявних проблем при вирішенні задачі позиціонування віртуальних об'єктів. Розглянуто алгоритми одночасної локалізації та картографування SLAM. Наведено практичні результати інтеграції алгоритмів SLAM для кращого визначення позицій віртуальних об'єктів та орієнтації мобільного пристрою у навколишньому середовищі.

2. Існуючі рішення формування доповненої реальності

Доповнена реальність являє собою звичайну реальність, в якій додана цифрова графіка. На відміну від віртуальної реальності, котра вимагає повного занурення у віртуальне створене середовище, доповнена реальність використовує середовище навколо нас та просто накладає поверх його зображення певну віртуальну інформацію, наприклад графіку, текст, 3D-моделі та реакцію на взаємодію з ними. Ця інформація використовується як додатковий корисний інструмент, що забезпечує допомогу в повсякденній реальності [1].

Для створення мобільних додатків з доповненою реальністю розроблено декілька бібліотек. Вони покривають різні необхідності розробників та мають певні особливості. Але більшість їх методів мають спільне призначення: розпізнавання цільових зображень та об'єктів у навколишньому середовищі, формування, позиціонування та відображення об'єктів доповненої реальності.

Рішення задачі позиціонування об'єктів доповненої реальності складається з послідовних етапів:

1) завантаження ознак цільових об'єктів. Є один або декілька маркерів (об'ємне тіло, картинка, текст тощо), котрі необхідно знайти у навколишньому середовищі. Для кожного з цих маркерів формується набір ознак, за якими можна виявити їх на вхідному зображенні. Цими ознаками можуть бути розміщення граней, точок, колір абощо;

2) попередня обробка зображення з камери. З певною частотою з відеопотоку камери витягується фрейм для обробки. Оскільки є зовнішні фактори, що впливають на чіткість картинки, попередня обробка включає в себе адаптацію яркості, контрасту та насиченості кольорів отриманого фрейму. В результаті з відеопотоку з певною частотою отримаємо по зображенню, підготовленому для подальших дій;

3) виявлення ознак маркерів на зображенні. Після попередньої обробки, фрейм передається алгоритму розпізнавання образів для співставлення ознак кожного маркеру на картинці. Якщо

маркер знайдено – фіксується його місцеположення на фреймі;

4) формування координатних осей для доповненої реальності. На основі позицій знайдених на фреймі маркерів формується система координат. При її формуванні враховується місцеположення кожного маркеру та кут нахилу камери до нього;

5) відображення об’єктів доповненої реальності (рендеринг). Створена система координат слугує основою для розміщення віртуальних об’єктів поверх зображення навколишнього середовища, отриманого з камери. Їх позиції відносно маркерів визначаються розробником, а кут нахилу до камери залежить від куту повороту координатних осей.

Найвідоміші бібліотеки для побудови доповненої реальності – це Vuforia, Wikitude, EasyAR [2]. У всіх них є функція розпізнання маркерів та відображення віртуальних об’єктів на їх місці. У якості віртуальних об’єктів можуть виступати: текстові написи, зображення, відео, графічні елементи, 3D-моделі.

3. Недоліки в роботі алгоритмів існуючих програмних бібліотек

У кожній з вищеназваних відомих бібліотек є розширений режим відслідковування маркерів. Цей режим частково дозволяє частково будувати доповнену реальність, орієнтуючись на об’єкти у навколишньому середовищі, що не були заздалегідь позначені розробником, як маркери.

У Vuforia [4] цей функціонал носить назву «розширене відслідковування» (Extended Tracking). Його суть у тому, що спочатку він будує доповнену реальність, орієнтуючись на певну мітку, а потім орієнтується у просторі, беручи до уваги переміщення камери відносно об’єктів навколишнього середовища, показники акселерометра та гіроскопа. Недолік цього підходу – для відображення віртуальних об’єктів завжди необхідно, щоб був початковий маркер (рис. 2). Також, якщо камера вийшла за межі усіх віртуальних об’єктів, бібліотека «губиться» у просторі та перестає відслідковування, допоки знову не розпізнає початковий маркер.

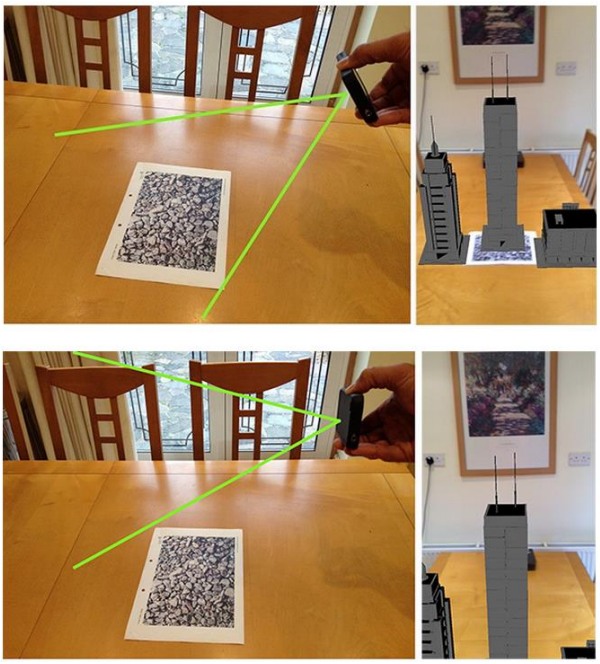


Рис. 1. Технологія Extended Tracking від Vuforia

У свою чергу Wikitude та EasyAR пропонують «миттєве відслідковування» (Instant Tracking) [3], такий собі псевдо-SLAM (рис. 2). Перед формуванням доповненої реальності, ці бібліотеки генерують мітки з навколишнього середовища. Відразу є їх розпізнають та відображають елементи доповненої реальності. При повороті камери бібліотека орієнтується у просторі, спираючись на ті ж згенеровані мітки. Це виглядає подібно до орієнтування без маркерів, та насправді маркери просто миттєво генеруються під час виконання програми.

Серед недоліків цього підходу потрібно зазначити, що цей підхід працює нестабільно у одноманітних середовищах, наприклад на фоні однокольорової стіни. Часто відбувається дезорієнтація, внаслідок чого об’єкти доповненої реальності позиціонуються невірно, або взагалі можуть зникнути з поля зору.

Рис. 2. Технологія Instant Tracking від Wikitude

4. Концепція SLAM

SLAM [7] (англ. simultaneous localization and mapping) є алгоритмічною обчислювальною задачею побудови і оновлення мапи невідомого оточення з одночасним відстеженням місцеположення у процесі руху. Алгоритми SLAM обмежуються наявними ресурсами, таким чином не можуть бути абсолютно досконалими, бо досягають оперативної доступності.

Концепція, загальна методологія SLAM полягаю у вирішенні двох завдань [6]:

1) побудова карти дослідженого простору;

2) побудова траєкторії руху робота на мап.

Перший принциповий момент – SLAM не припускає будь-яких знань про середовище – ні міток на місцевості, ні попередньої карти немає. Всі рішення будуються тільки на результатах вимірювань датчиків (зазвичай це rgb- і далекомірна камери, але, буває, і додаткове обладнання гіроскопа, gps-датчика і т.д.).

Другий не менш важливий момент – середовище вважається статичним. Це означає, що якщо робот проїхав повз фікуса, ніхто за його спиною цей же самий фікус рухати не стане. В кадрі жодного стороннього руху немає – ніхто в кадрі не ходить. Освітлення радикально не змінюється – тобто тіні по стінах не стрибають. Такі умови складно гарантувати в природному середовищі – шелестить листя або кішка запросто може дезорієнтувати робота в просторі.

Постановка задачі виглядає наступним чином:

Дана послідовність даних спостереження сенсору o_t за дискретні проміжки часу t, задачею SLAM є розрахувати і визначити розташування агента x_t і мапу оточення m_t. Всі величини зазвичай ймовірнісні, тому необхідно обчислити:

$$P(x_t|o_{1:t}, m_t).$$

Застосування формули Баєса дає основу для послідовного оновлення апостеріорного розташування, при даній мапі і функції переходу $P(x_t, x_{t-1})$

$$P(x_t|o_{1:t}, m_t) = \sum_{m_{t-1}} P(o_t|x_t, m_t)\, L_t,$$

$$L_t = \sum_{x_{t-1}} \frac{P(x_t|x_{t-1})P(x_{t-1}|m_t, o_{1:t-1})}{Z}. \quad (1)$$

Аналогічно, мапа може оновлюватися послідовно наступним шляхом:

$$P(m_t|x_{t-1}, o_{1:t}) = \sum_{x_t}\sum_{m_t} P(m_t|x_t, m_{t-1,} o_t)\, K_{t,}$$

$$K_t = P(m_{t-1}, x_t|o_{1:t-1}, m_{t-1}). \quad (2)$$

Як для більшості задач наближення, рішення можна знайти при наближенні двох змінних, до локального оптимального рішення, шляхом почергового оновлення двох рівнянь у формі ЕМ-алгоритму.

У загальному випадку SLAM можна описати як послідовність кроків [7]:

1) сканування навколишнього простору;

2) визначення зміщення на основі порівняння поточного кадру з попереднім;

3) виділення на поточному кадрі особливостей-міток;

4) зіставлення міток поточного кадру з мітками, отриманими за всю історію спостережень;

5) оновлення на основі цієї інформації положення пристрою за всю історію спостережень;

6) перевірка на петлі – чи не проходимо ми повторно по одній і тій же місцевості;

7) вирівнювання загальної карти світу (відштовхуючись від положення міток і пристрою за всю історію спостережень).

До статистичних методів, що використовуються для задач апроксимації наведених вище, належать: фільтр Калмана, фільтр часток (який є методом Монте-Карло) і узгоджене сканування діапазонних даних. Вони дозволяють визначити оцінку функції апостеріорної ймовірності для позиції пристрою і параметрів мапи. Техніки оцінювання приналежності до множини в основному засновуються на поширенні сталого інтервалу. Вони забезпечують множину, яка містить позицію пристрою і множину апроксимації мапи.

5. Практичне застосування алгоритму SLAM у побудові доповненої реальності

Для досліджень результатів застосування алгоритму SLAM у побудові доповненої реальності було проведено експеримент на відкритій вуличній місцевості. У якості розміщуваного об’єкта доповненої реальності було побудовано паралелепіпед, розфарбований градієнтом для кращого виділення граней та ребер.

Користувачем було пройдено 3 різні маршрути, наведені на рис. 3.

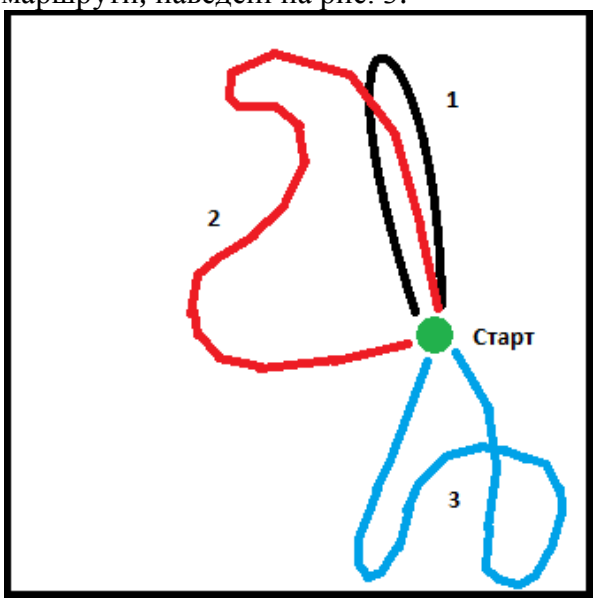


Рис. 3. Схема пройдених маршрутів

Для прикладу розглянемо результати виконання алгоритму при проходженні третього маршруту. При повернені на початкову точку алгоритм доповнив реальність з точністю до 20 см. У початковій точці зображено паралелепіпед (рис. 4), після чого пройдено маршрут №3. Результат можна побачити на рис. 5.

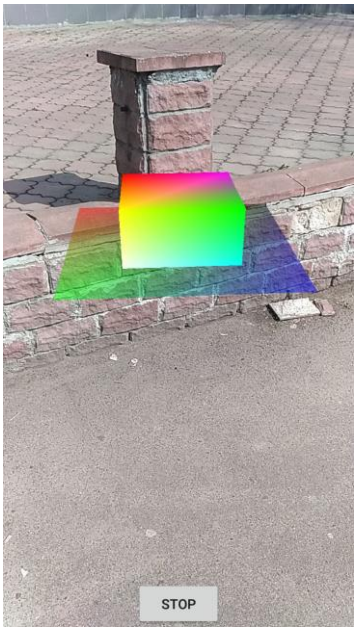


Рис. 4 Початкова точка до проходження маршруту

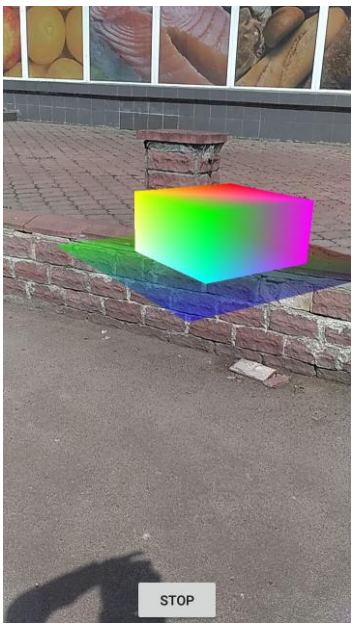


Рис. 5. Початкова точка після проходження маршруту

Для порівняння результатів виконання алгоритму, було враховано кількість поворотів у маршруті, його довжину та точність позиціонування об’єктів доповненої реальності (табл. 1).

Табл. 1. Результати експериментів. Точність позиціонування для різних маршрутів.

Маршрут	К-сть поворотів, шт.	Довжина, м	Точність, см
1	1	5	13
2	3	9	19
3	3	8	20

6. Результати дослідження

Було проведено дослідження застосування алгоритмів одночасної локалізації та картографування SLAM при формуванні доповненої реальності. Існуючі алгоритми побудови доповненої реальності використовують маркери для орієнтування у навколишньому середовищі. Це накладає певні обмеження та викликає нестабільну роботу при роботі з одноманітними поверхнями. Було реалізовано алгоритм без застосування маркерів. Було проведено експериментальне дослідження можливості застосування алгоритму та виявлено, що при повернені на початкову точку алгоритм доповнив реальність з точністю до 20см.

Список літератури

1. Обзор средств реализации AR приложений [Електрон. ресурс]. – Режим доступу: http://nppsatek.com/blogs/2018/01/10/
2. Best augmented reality SDK for AR development for iOS and Android [Електрон. ресурс]. – Режим доступу: https://thinkmobiles.com/blog/best-ar-sdk-review/
3. 8 лучших SDK дополненной реальности для iOS и Android в 2017 году [Електрон. ресурс]. – Режим доступу: https://holographica.space/articles/8-best-ar-sdk-2017-9287
4. Інтерфейси доповненої реальності Vuforia Engine [Електрон. ресурс]. – Режим доступу: http://dspace.pnpu.edu.ua/bitstream/123456789/5176/1/Prokopenko.pdf
5. Søren Riisgaard and Morten Rufus Blas «SLAM for dummies» 2015. – 127 с.
6. Сойфер В.А. «Методы компьютерной обработки изображений (2-е издание)» М.: Физматлит, 2003. – 782 с.
7. SLAM – принципы и ссылки на open source [Електрон. ресурс]. – Режим доступу: http://my-it-notes.com/2013/01/slam-basis-and-links-at-open-source/

УДК 004.94; 004.4; 004.62

КОВАЛЬ А.А.
ЖАРІКОВ Е.В.,

МЕТОД РОЗМІЩЕННЯ ВІРТУАЛЬНИХ МАШИН НА ОСНОВІ НАВЧАННЯ З ПІДКРІПЛЕННЯМ

Запропоновано метод розміщення віртуальних машин на основі навчання з підкріпленням, який при виборі управляючих впливів враховує витрати електроенергії та час порушення вимог угоди про рівень послуг. Розроблений алгоритм агента, який враховує зміни робочого навантаження на ресурси для прийняття рішення щодо включення або переключення в сплячий режим незавантажених фізичних серверів з метою зменшення витрат електроенергії. Запропонований агент навчання з підкріпленням базується на підході *Q*-навчання, який дозволяє визначати наближену до оптимальної політику управління режимами роботи фізичного сервера без створення моделі середовища та попередньої інформації про навантаження. Виконано програмну реалізацію запропонованого методу.

The method of dynamic virtual machine placement based on reinforcement learning is proposed. The proposed method takes into account the power consumption and the time of SLA violations while producing control impacts. The proposed agent's algorithm takes into account the changes in the resource utilizations to make a decision on whether or not to switch underloaded physical servers to the sleep mode in order to reduce the power consumption. The proposed reinforcement learning agent is based on the *Q*-learning approach which allows to determine the optimal policy for managing the physical servers without creating an environment model and preliminary information about the workload. Software implementation of the proposed method is implemented.

Ключові слова: центр обробки даних, навчання з підкріпленням, віртуалізація, енергоефективність, SLA

1. Вступ

Реалізація інформаційних процесів та надання доступу до сучасних інформаційних сервісів пов'язані із використанням центрів обробки даних (ЦОД), або дата-центрів – комплексних централізованих систем для створення високопродуктивної, відмовостійкої IT-інфраструктури. До головних завдань ЦОД належать консолідоване зберігання і опрацювання даних користувачів, підтримка функціонування застосунків та надання користувачам прикладних сервісів на заданому якісному рівні.

Ефективне управління ЦОД пов'язане з необхідністю розв'язання низки проблем, насамперед створення умов для функціонування інформаційно-обчислювальних потужностей ЦОД, управління віртуалізованими ресурсами, забезпечення надійності та безпеки. Вкладаючи кошти, компанії намагаються збільшити прибуток, покращити якість сервісів, зменшити витрати на експлуатацію ЦОД, знизити вартість обслуговування користувачів, що дозволить, зрештою, закласти основу для ефективної діяльності як самої компанії, так і клієнтів.

Забезпечення рівня вимог користувачів з мінімізацією витрат становить сутність проблеми управління функціонуванням ЦОД. Зазвичай цю комплексну проблему розбивають на ряд підзадач. Однією з них є задача управління ресурсами і навантаженням ЦОД.

Необхідні гнучкі рішення, які ґрунтуються на оцінюванні стану ресурсів і обсягів навантаження та полягають у правильному розподілі навантаження і ефективному управлінні ресурсами. Для систематичного прийняття правильних рішень необхідні інструментарій та комплекс методик і алгоритмів для вирішення задач управління IT-інфраструктурою. Їх створення становить важливу науково-практичну проблему, розв'язання якої вимагає розуміння процесів, які відбуваються в хостингових компаніях, функціонування IT-інфраструктури, чіткої постановки конкретних завдань дослідження, розробки математичних

моделей, моделей ЦОД і відповідних методів вирішення задачі та реалізації згаданих вище методик і алгоритмів.

Технології віртуалізації надають можливість абстрагуватися від особливостей окремих груп ресурсів, об'єднати їх у апаратно-програмні комплекси потрібної конфігурації і спростити управління ними. Віртуалізація ресурсів інфраструктури полягає у створенні віртуальних машин (ВМ) – програмних абстракцій, що запускаються на платформі фізичних апаратно-програмних (хостових) систем [1].

В роботі пропонується метод розміщення ВМ на основі навчання з підкріпленням (НП, англ. reinforcement learning, RL) [2]. НП – один із способів машинного навчання, в ході якого програмні агенти виконують дії в певному середовищі з метою максимізації сукупної винагороди як результату дій. В дискретні моменти часу агент отримує спостереження, яке включає і винагороду, та обирає керуючий вплив з множини доступних впливів на середовище. Середовище переходить до нового стану, і визначається винагорода, пов'язана з переходом в новий стан. Метою агента НП є отримання якомога більшої винагороди. НП є добре пристосованим для задач, які включають компроміс між довготерміновою та короткотерміновою винагородою. НП відрізняється від стандартного навчання з учителем тим, що пари правильних входів/виходів ніколи не представляються, а неоптимальні дії явно не виправляються.

2. Постановка задачі

Модель ЦОД, як постачальник віртуальних ресурсів, складається з m фізичних серверів (ФС) з різною конфігурацією. Кожен ФС має процесор, який може бути багатоядерним, при цьому продуктивність ядер вимірюється в мільйонах інструкцій на секунду (англ. Million instructions per second, MIPS). Крім того, ФС характеризується обчислювальною потужністю процесора, об'ємом оперативної пам'яті, пропускною здатністю мережі та ємністю жорсткого диску. Користувачі відсилають завдання або запити для створення n ВМ. Спочатку ВМ розміщуються на ФС відповідно до запитуваних характеристик. Відсоток використання ресурсів ВМ з часом буде змінюватись. Необхідно розмістити ВМ на мінімальній кількості ФС для зменшення витрат електроенергії та часу порушення вимог угоди про рівень послуг (англ. Service-level agreement, SLA).

3. Метод розміщення ВМ на основі навчання з підкріпленням

Модель ЦОД та агента навчання з підкріпленням наведена в [3].

Метою запропонованого методу є зменшення витрат електроенергії в ЦОД та зменшення часу порушення вимог SLA. SLA включає в себе вимоги та обмеження щодо якості послуг, що надаються провайдером: часу відгуку, доступності, пропускної здатності, безпеки та ін. SLA виконується для кожного клієнта, коли вся продуктивність, яку потребують застосунки всередині ВМ, забезпечується в будь-який час.

ВМ розміщуються на ФС відповідно до поточних затребуваних ресурсів. Коли використання ресурсів на ФС є низьким, всі ВМ перерозподіляються на інші ФС, а недостатньо завантажений ФС переходить до сплячого режиму. Крім того, коли ФС перевантажений, деякі ВМ повинні бути переміщені, щоб зменшити час порушення вимог SLA. Якість алгоритму розміщення ВМ може поліпшуватися в процесі роботи агента НП та алгоритму Q-навчання для визначення режиму роботи кожного ФС (сплячий або активний).

Алгоритм може динамічно адаптувати ФС до зміни робочого навантаження. Важливою частиною алгоритму є вирішення питання про те, чи додатковий ФС повинен забезпечити ефективне використання ресурсів зі збільшенням робочого навантаження, або резервні ФС можуть бути переведені в сплячий режим або чи достатня поточна кількість ФС. Агент НП вважається важливою частиною алгоритму для прийняття такого рішення.

ФС, які задіяні в процесі оптимізації, позначаються наступними мітками:

– HIBERNATE – позначаються ФС, які потрібно перевести до сплячого режиму;

– PLACEMENT – позначаються ФС, які задіяні в процесі розміщення ВМ;
– AVAILABLE – позначаються ФС, які доступні для процесу розміщення ВМ.

Алгоритм розміщення ВМ на основі НП наведено на рисунку 1.

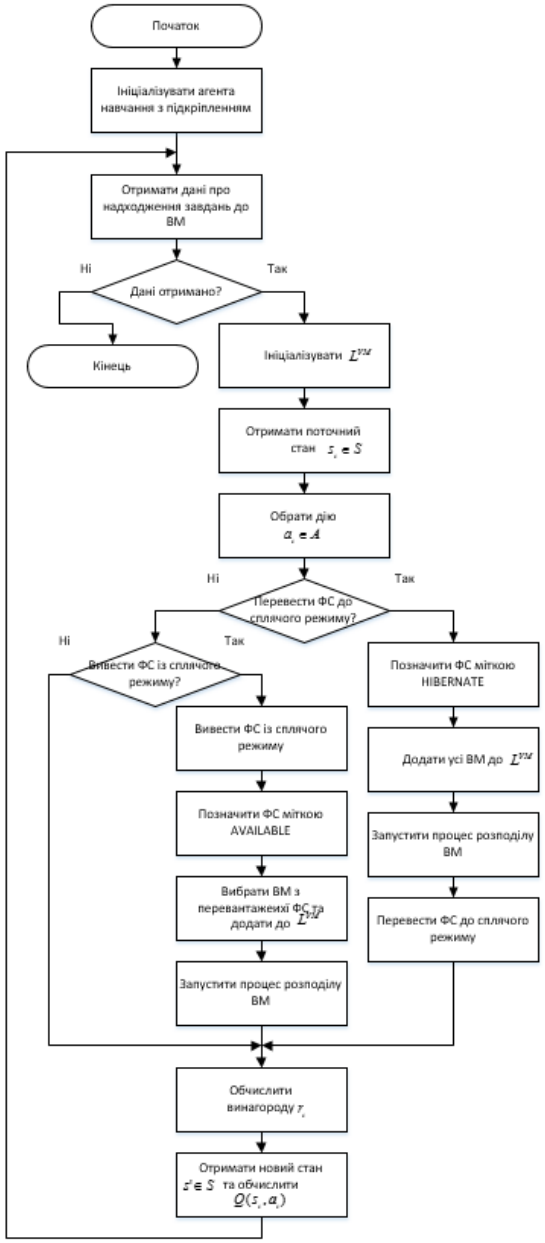


Рис. 1. Алгоритм розміщення ВМ на основі НП

Алгоритм розподілу ВМ наведено на рисунку 2. На вхід алгоритм отримує список L^{VM} з ВМ, які треба розподілити по ФС. Результатом роботи алгоритму є карта міграцій ВМ до ФС в ЦОД.

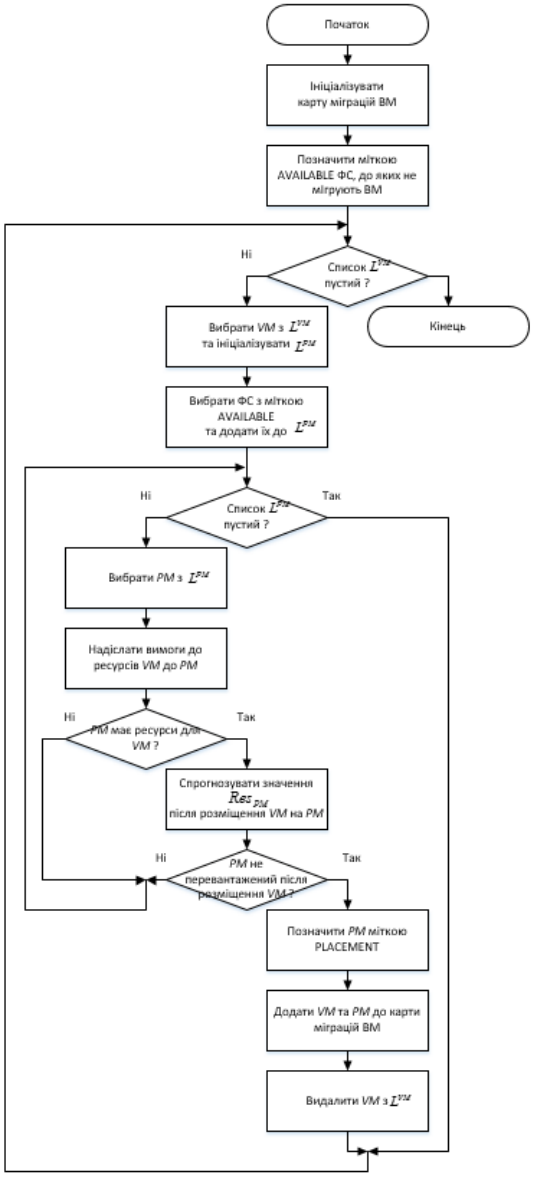


Рис. 2. Алгоритм розподілу ВМ

Виконується декілька перевірок перед тим, як ФС підтвердить можливість розміщення ВМ. Повинна виконуватись вимога (1) для кожного ресурсу:

$$Res_{PM}^{AVAIL} > Res_{VM},$$

де Res_{PM}^{AVAIL} – доступна кількість ресурсів ФС;

Res_{VM} – вимоги до кількості ресурсів з боку ВМ.

ФС починає процес розміщення ВМ, якщо виконується вимога (2) для кожного ресурсу:

$$Res_{PM}^{MAX} > Res_{PM}^{P},$$

де Res_{PM}^{MAX} – максимальна кількість ресурсів ФС;

Res_{PM}^{P} – прогнозоване навантаження.

Рівняння (3) визначає поточну ємність ЦОД, яка використовується для визначення динаміки використання ресурсів:

$$CAP_{Res}^{DC} = \frac{\sum_{j=1}^{n} Res_{VM}^{j}}{\sum_{i=1}^{m} Res_{PM}^{i}}, \quad (3)$$

де n – кількість ВМ в певний проміжок часу;
m – кількість ФС;
Res_{VM}^{j} – вимоги до кількості ресурсів j-ї ВМ;
Res_{PM}^{i} –поточне використання ресурсів i-го ФС.

4. Результати досліджень

Для проведення дослідження було використано фреймворк для моделювання хмарної інфраструктури та сервісів CloudSim 4.0 [4]. Вхідними даними є показники продуктивності ВМ, а саме показники використання процесора, оперативної пам'яті, мережі, які були зібрані з 500 ВМ розподіленого ЦОД Bitbrains [5] протягом одного місяця.

У таблиці 1 наведено ФС для моделювання кластеру хмарного ЦОД та їх споживання електроенергії у ватах при різних показниках завантаження процесора [6].

Табл. 1. Споживання електроенергії серверами при різних показниках завантаження процесора

ФС	0%	10%	20%	30%	40%	50%	60%	70%	80%	90%	100%
Dell Inc. PowerEdge R640	55	125	151	176	202	232	261	301	357	421	469
Dell Inc. PowerEdge R740	52.3	129	147	170	195	224	255	292	344	408	457
Dell Inc. PowerEdge R830	86.4	181	215	253	287	315	340	377	421	483	562
Dell Inc. PowerEdge R940	106	245	292	336	383	437	502	583	694	820	915

Оцінка ефективності методу розміщення ВМ на основі НП проводилася за трьома показниками: час порушення вимог SLA, споживання електроенергії та кількість міграцій. На рисунку 3 наведені графіки показників ефективності для різних значень швидкості навчання та коефіцієнта знецінювання.

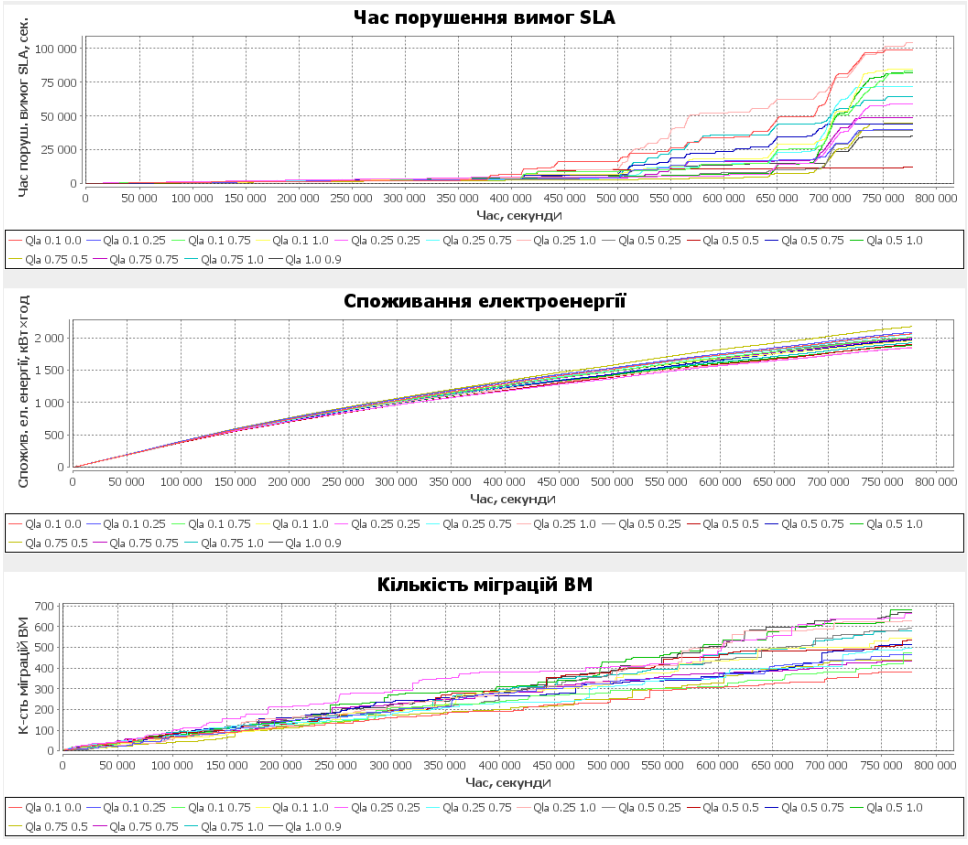


Рис. 3. Показники ефективності методу для різних значень швидкості навчання та коефіцієнта знецінювання

Беручи до уваги пріоритетність показника часу порушення вимог SLA, для подальшого дослідження виберемо $\alpha = 0.5$ та $\gamma = 0.5$, які забезпечують мінімальне значення порушення вимог SLA. На рисунку 4 наведені графіки показників ефективності для різних значень ваг, що визначають відносну важливість штрафу за порушення SLA та штрафу за споживання електроенергії.

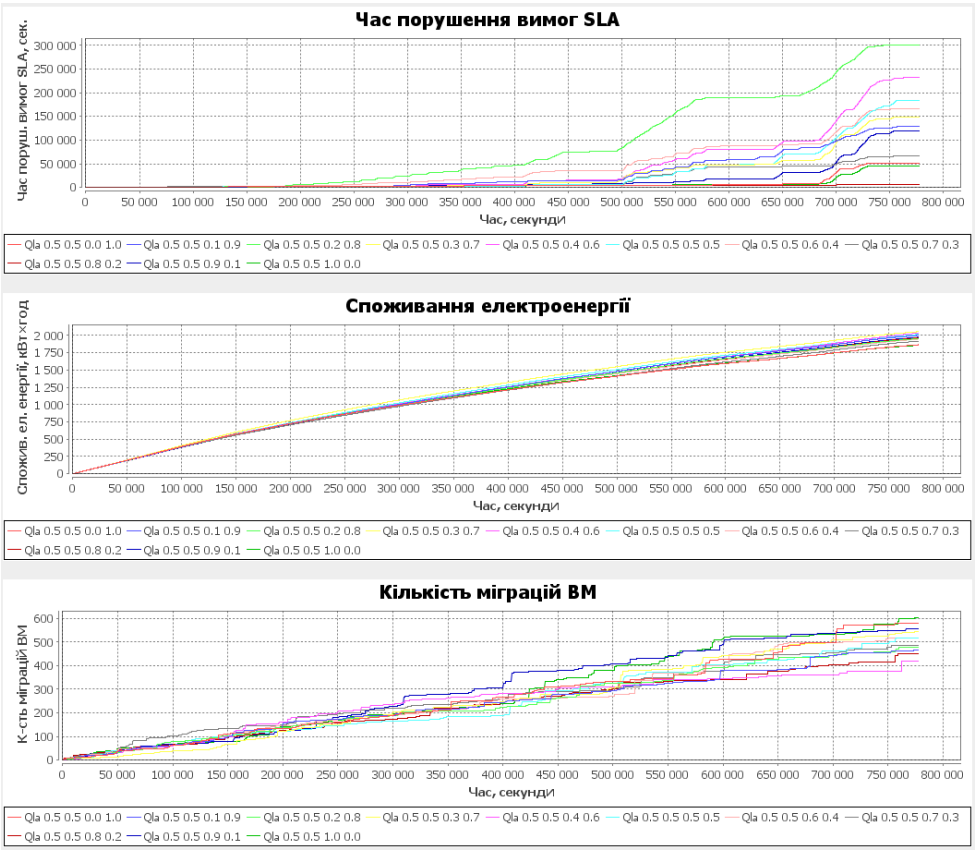


Рис. 4. Показники ефективності методу для різних значень ваг, що визначають відносну важливість штрафу за порушення SLA та штрафу за споживання електроенергії

Порівняємо отримані результати з алгоритмом динамічного розміщення ВМ LR-MMT-1.2 [7] (LR, Local Regression – політика вибору перевантажених ФС; MMT, Minimum Migration Time – політика вибору ВМ для міграції), який значно перевершує інші динамічні алгоритми розміщення ВМ (рисунок 5).

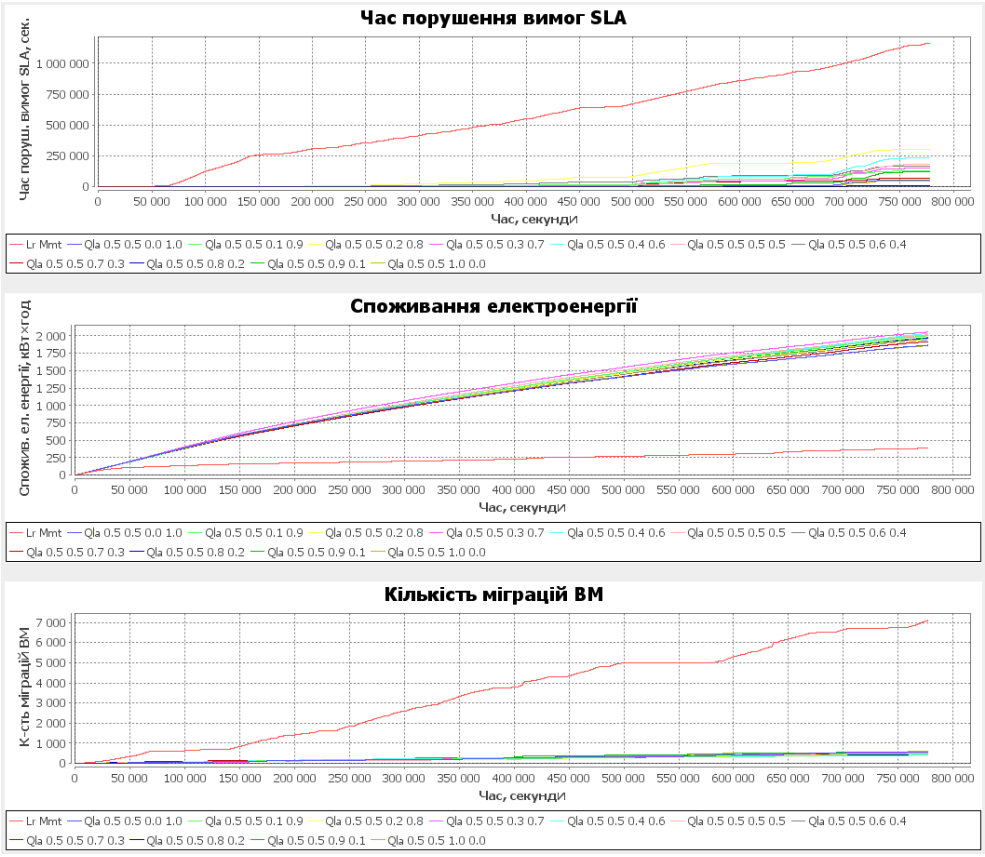


Рис. 5. Порівняння методу розміщення ВМ на основі НП з алгоритмом динамічного розміщення ВМ LR-MMT-1.2

За показниками часу порушення вимог SLA та кількості міграцій метод розміщення ВМ на основі НП є ефективнішим ніж алгоритм LR-MMT-1.2. Алгоритм LR-MMT-1.2 ефективніший за показником споживання електроенергії за рахунок великої кількості міграцій ВМ, що є неприпустимим в реальних ЦОД.

5. Висновок

Проаналізовано можливість застосування методу навчання з підкріпленням для управління ресурсами хмарних ЦОД. Запропоновано метод розміщення віртуальних машин на основі навчання з підкріпленням, який при виборі управляючих впливів враховує витрати електроенергії та кількість порушень SLA. Перевагою методу розміщення ВМ є здатність виконувати в режимі онлайн розміщення нових віртуальних машин одночасно з перерозподілом вже працюючих віртуальних машин.

Розроблений алгоритм агента, який, через сприймання стану фізичних серверів та віртуальних машин, враховує зміну робочого навантаження на ресурси для прийняття рішення щодо включення або переключення в сплячий режим незавантажених фізичних серверів з метою зменшення витрат електроенергії. Запропонований агент навчання з підкріпленням базується на підході *Q*-навчання, який дозволяє визначати наближену до оптимальної політику управління режимами роботи фізичного сервера без створення моделі середовища та попередньої інформації про навантаження. Запропоновані моделі використовуються для моделювання процесів управління ресурсами хмарного ЦОД.

Програмно реалізований метод розміщення віртуальних машин на основі навчання з підкріпленням виявився ефективнішим за алгоритм динамічного розміщення віртуальних машин LR-MMT-1.2 за показниками часу порушення вимог SLA та кількості міграцій.

Список літератури

1. Теленик С.Ф. Управління навантаженням і ресурсами центрів оброблення даних при виділених серверах [Текст]: /С.Ф. Теленик, О.І. Ролік, М.М. Букасов, Р.В. Римар, К. О. Ролік.– Автоматика. Автоматизація. Електротехнічні комплексі та системі. – 2009. – №2 (24). – С.122 – 136.
2. Sutton R.S. Reinforcement Learning: An Introduction [Текст]: /R.S.Sutton, A.G.Barto //MIT Press.– 1998. – c.360
3. Жаріков Е. В. Динамічне розміщення віртуальних машин на основі навчання з підкріпленням в хмарних центрах обробки даних / Е. В. Жаріков, А. А. Коваль, Р. А. Терентьєв. // Наукові вісті Далівського університету. - 2017. - № 13.
4. CloudSim: A Framework For Modeling And Simulation Of Cloud Computing Infrastructures And Services [Електронний ресурс] // Режим доступу: http://www.cloudbus.org/cloudsim/.
5. GWA-T-12 Bitbrains [Електронний ресурс] // Режим доступу: http://gwa.ewi.tudelft.nl/datasets/gwa-t-12-bitbrains.
6. Standard Performance Evaluation Corporation. SPECpower_ssj2008 Results [Електронний ресурс] // Режим доступу: http://spec.org/power_ssj2008/results/.
7. A. Beloglazov, R. Buyya, “Optimal online deterministic algorithms and adaptive heuristics for energy and performance efficient dynamic consolidation of virtual machines in Cloud data centers.” Concurrency and Computation: Practice and Experience, vol. 24, no. 13, pp. 1397-1420, 2012.

УДК 004.93(015.7)

КІНДЗЕРСЬКИЙ О.В.
ОЛІЙНИК Ю.О.

РЕАЛІЗАЦІЯ АЛГОРИТМУ КЛАСТЕРИЗАЦІЇ ДАНИХ K-MEANS НА ОСНОВІ ТЕХНОЛОГІЇ NVIDIA CUDA

В даній статті розглянуто одну із задач інтелектуального аналізу даних – кластеризацію. Запропоновано використання технології Nvidia CUDA для паралелізації обраного алгоритму. Описано та реалізовано алгоритм k-means з використанням .Net фреймворка CUDAfy.Net для роботи з графічними процесорами Nvidia. Проаналізовано результати швидкодії модифікованого алгоритму кластеризації k-means на графічному процесорі(GPU) в порівнянні з центральним процесором(CPU).

The subject of the article is one of the data mining tasks – cluster analysis. An application of Nvidia CUDA technology for parallelization of chosen algorithm is proposed. The k-means algorithm using .Net framework CUDAfy.Net for Nvidia GPUs cooperation is described and implemented. The performance results of clustering using k-means algorithm are analyzed for graphic (GPU) and central (CPU) processors.

1. Вступ

Задача кластеризації даних є важливою складовою інтелектуального аналізу даних(ІАД). Завдання кластеризації на відміну від інших задач ІАД передбачає навчання без учителя, тобто надає можливість розподілити наявні об’єкти на класи на підставі подібності їх характеристик. Кластеризація забезпечує краще розуміння об’єктів, що досліджуються, та спростити подальший їх аналіз. Оскільки загальний об’єм даних на 2012 рік складав більше 1,8 зеттабайт (1,8 трлн Гб) та за дослідженнями IDC (International Data Corporation) ця цифра подвоюється кожні 2 роки[3], кластеризація застосовується для стиснення даних. Це забезпечується тим, що об’єкти, які об’єднанні в кластер, можна розглядати як один об’єкт для подальшого опрацювання. Оскільки ми розглядаємо опрацювання надвеликих масивів даних доцільно використовувати усі можливі засоби для прискорення обчислень. Одним з найпопулярніших і найефективнішим методів є звичайно розпаралелювання обчислювальних процесів. Одним варіантом є використання розподілених систем, що є досить складним і вимагає додаткових затрат на устаткування. В той же час користувачі мають змогу використовувати графічні процесори Nvidia, що підтримують технологію CUDA, яка має високу продуктивність паралельних обчислень.

CUDA(англ. Compute Unified Device Architecture) – програмно-апаратна архітектура паралельних обчислень, яка дозволяє істотно збільшити обчислювальну продуктивність завдяки використанню графічних процесорів фірми Nvidia[2]. Платформа паралельних обчислень CUDA забезпечує набір розширень для мов C і C++, що дозволяють висловлювати як паралелізм даних, так і паралелізм завдань на рівні дрібних і великих структурних одиниць. Програміст може вибрати засоби розробки: мови високого рівня, такі як C, C++, Fortran або ж відкриті стандарти, такі як директиви OpenACC. Для мови високого рівня C# був розроблений фреймворк CUDAfy .Net, який дає змогу розробляти програмне забезпечення з великим показником продуктивності і використанням технології Nvidia CUDA для Microsoft .Net Framework[4].

2. Формальна постановка задачі кластеризації

Дано набір даних з наступними властивостями:

- Кожний екземпляр даних виражається однаковою кількістю параметрів з чітким числовим значення;

- Клас для кожного конкретного екземпляру даних до початку дослідження невідомий.

Потрібно знайти:

- Спосіб порівняння даних між собою(міру схожості);
- Спосіб кластеризації;
- Розбиття даних на кластери

Формально задача кластеризації описується наступним чином. Дано множину об'єктів даних I, кожен з яких представлений набором атрибутів. Потрібно побудувати множину кластерів C і відображення F множини I на множину C, F: I -> C. Відображення F задає модель даних, що являється власне вирішенням задачі[1].

Множина I визначається наступним чином

$$I = \{i_1, i_2, \ldots, i_j, \ldots, i_n\}, \quad (1)$$

де i_j – об'єкт, що досліджується.

Кожен із об'єктів характеризується набором параметрів:

$$i_j = \{x_1, x_2, \ldots, x_h, \ldots, x_m\} \quad (2)$$

Задача кластеризації полягає в побудові множини:

$$C = \{c_1, c_2, \ldots, c_k, \ldots, x_g\}, \quad (3)$$

де c_k – кластер, який має в собі похожі один на одного об'єкти з множини I:

$$c_k = \{i_l, i_p \mid i_l \in I, i_p \in I, d(i_l, i_p) < \sigma\}, \quad (4)$$

де σ – величина, яка визначає міру близькості для включення об'єктів в один кластер, $d(i_l, i_p)$ – відповідно міра близькості.

Міру близькості будемо розраховувати за допомогою метрики Евкліда, де відстань обчислюється наступним чином:

$$d(x_i, x_j) = \sqrt{\sum_{t=1}^{m} (x_{it} - x_{jt})^2} \quad (5)$$

3. Модифікація алгоритму k-means

Алгоритм k-середніх був запропонований ще в 1979 році і він не втрачає своєї актуальності і сьогодні. Даний алгоритм являється прообразом практично всіх алгоритмів нечіткої кластеризації, і його розгляд допоможе нам краще зрозуміти принципи, що закладені в більш складні алгоритми.

Сьогодні використовуються різні модифікації даного алгоритму, але кожен із них має два основних недоліки: необхідність знати кількість кластерів перед виконанням алгоритму і оптимальність отриманих рішення не є гарантованою.

Розроблений алгоритм представляє собою ітераційну процедуру:

- Крок 1 (CPU): Випадковим чином обрати центри кластерів із елементів вхідних даних, встановити номер ітерації $l = 0$.
- Крок 2 (CPU): Підключити GPU модуль, обрахувати конфігурацію запуску для функції-ядра:
  - Кількість ниток(так прийнято називати один потік виконання обрахунків) в блоці:

$$numT = \frac{maxT}{2}, \quad (6)$$

де $maxT$ – максимальна можлива кількість ниток в блоці для підключеного графічного процесора;

  - Кількість блоків:

$$numB = \left\lceil \frac{num0}{numT} \right\rceil, \quad (7)$$

де $num0$ – загальна кількість об'єктів, що досліджується, $numT$ – визначена кількість ниток в блоці.

В оригінальному алгоритмі наступні два кроки полягають у обчисленні матриці розбиття за формулою 8,та у визначенні нових центрів кластерів за формулою 9. В нашій модифікації суть даних кроків зберігається, але присутня специфічна реалізація в зв'язку з застосуванням паралельних обчислень.

$$u_{ij}^{(l)} = \begin{cases} 1, \text{при } d(m_j, c_i) = \min\limits_{l \le k \le c} d(m_j, c_k) \\ 0, \quad \text{в інших виппадках} \end{cases} \quad (8)$$

$$c_l^{(i)} = \frac{\sum_{j=1}^{d} u_{ij} m_j}{\sum_{j=1}^{d} u_{ij} m_j}, 1 \le i \le c \quad (9)$$

- Крок 3 (GPU): Викликати кернел-функцію з конфігурацією запуску, обчисленою на кроці 2, для опрацювання даних на графічному процесорі. Кернел-функція виконується паралельно на великій

кількості ниток, при чому кожна нитка опрацьовує один об'єкт із вхідної послідовності, що досліджується:

- В спільний пам'яті виділити масиви пам'яті для збереження накопичувальної сума суми характеристик об'єктів(*SharedSums*), що належать для певного кластера та відповідно кількості об'єктів в кожному кластері(*SharedCounts*);
- Синхронізувати нитки за допомогою бар'єрної синхронізації;
- Знайти оптимальний центроїд (кластер) для об'єкта, що досліджується;
- Додати характеристики об'єкта до масиву *SharedSums* відповідно до оптимального центроїда;
- Інкрементувати кількість об'єктів в оптимальному кластері на 1;
- Синхронізувати нитки за допомогою бар'єрної синхронізації;
- Копіювати дані масивів з спільної пам'яті в результуючі масиви в глобальній пам'яті;

Коментар: таким чином на графічному процесорі ми паралельно обчислили одразу потрібні нам:

$SharedSums_i = \sum_{j=1}^{d} u_{ij} m_j$ та

$SharedCounts_i = \sum_{j=1}^{d} u_{ij} m_j$

з формули 9 для кожного кластера, де $1 \le i \le c$

- Крок 4 (CPU): Визначити нові центри кластерів:

$$c_l^{(i)} = \frac{SharedSums_i}{SharedCounts_i}, 1 \le i \le c \quad (10)$$

- Крок 5: Перевірити наскільки змістилися центри кластерів. Якщо вони змістилися менше ніж на обрану точність δ – завершити процес, якщо ні – перейти до кроку 2 з номером ітерації *l = l + 1*.

Також існує набір обмежень: $u_{ij}^{(l)} \{0,1\}; \sum_{j=1}^{c} u_{ij} = 1; 0 < \sum_{j=1}^{d} u_{ij} < d,$ який визначає, що кожен вектор даних може належати тільки одному кластеру і не належати іншим. В кожному кластері повинно бути не менше одного елемента даних і не більше загальної кількості елементів.

4. Результати дослідження

Для порівняння виконання алгоритмів було згенеровано 10000 двовимірних точок. Для того щоб уникнути генерації різних псевдовипадкових чисел в оригінальному і модифікованому алгоритмах був встановлений однаковий seed для генератора випадкових чисел.

Так на Рис.1 ми спостерігаємо розбиття вхідних точок на 10 кластерів, що відображаються різними кольорами, а також в інформаційній в правій частині вікна, що обидва алгоритми виконалися за однакову кількість ітерації – 93, але виконання на CPU зайняло 1433 мс, а виконання на GPU – 27 мс. Це показує нам, що навіть на невеликому наборі простих даних модифікований алгоритм дає нам чималий виграш в його швидкодії.

В таблиці 1 наведено порівняльну характеристику часу кластеризації на 16 кластерів оригінального і модифікованого алгоритмів для двовимірних точок кількістю від 3000 до 10000 з кроком в 1000. Для тестування використовувався персональний комп'ютерний обладнаний CPU: Intel(R) Core(TM) i5-6500 CPU @ 3.20GHz та GPU: Nvidia GeForce GTX 960.

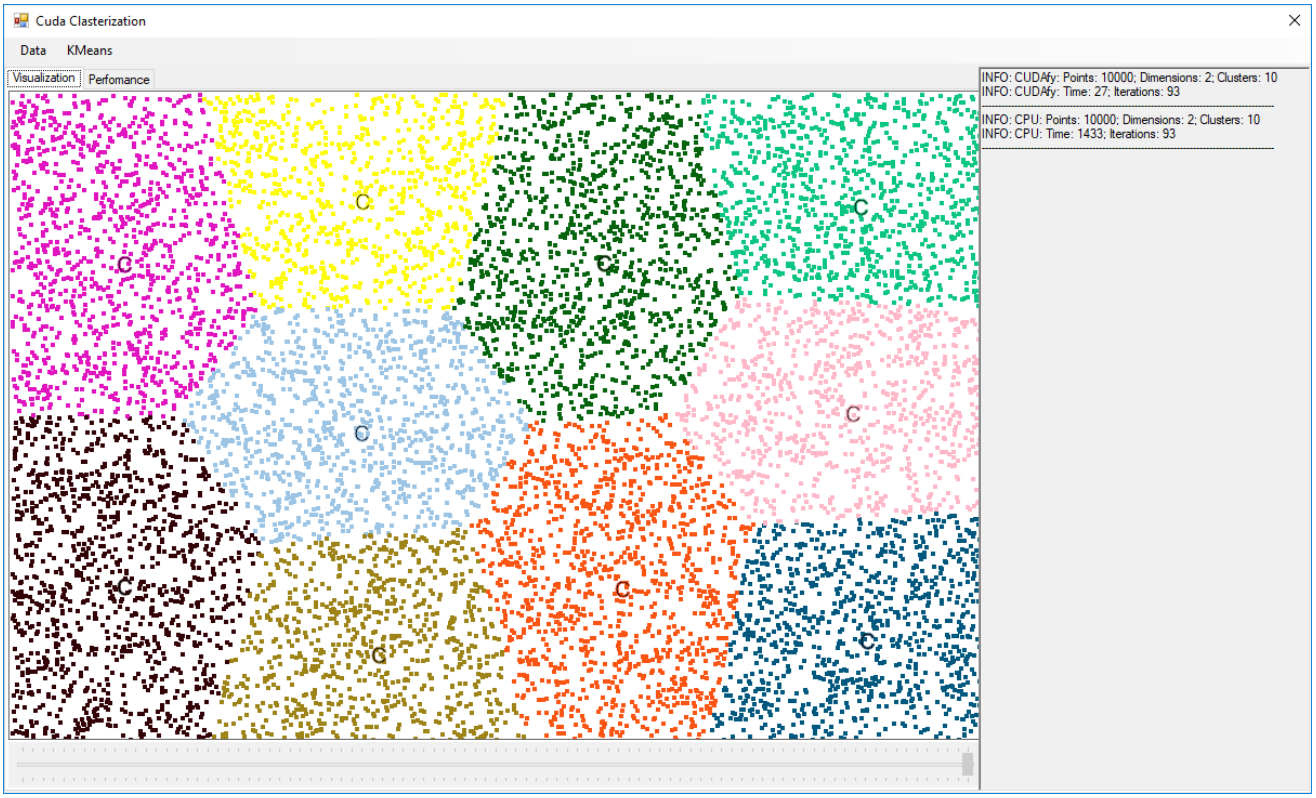


Рис. 1. Результати роботи алгоритму для 10 000 вхідних точок

Табл. 1. Порівняння швидкодії роботи алгоритму на CPU і CPU+GPU.

Кількість точок	К-сть ітерації	CPU, мс	CPU + GPU, мс	Коефіцієнт прискорення
3000	42	306	11	27,8
4000	61	593	15	39,5
5000	57	568	15	37,9
6000	61	890	17	52,4
7000	60	1014	16	63,4
8000	81	1593	25	63,7
9000	65	1414	22	64,3
10 000	79	1910	29	65,9

5. Висновки

З отриманих результатів можна зробити висновок, що використання технології Nvidia CUDA покращує алгоритм k-means для кластерного аналізу і дає великий приріст в швидкодії даного алгоритму, що видно в таблиці 1. Варто підмітити, що час виконання модифікованого алгоритму в більшій мірі залежить від збіжності алгоритму, тобто кількості ітерацій, а не від кількості вхідних точок. Це забезпечується власне масово паралельними обчисленнями на графічному процесорі, оскільки створюється кількість ниток відповідно до кількості вхідних точок і одночасно виконується обчислення для всіх вхідних об’єктів в своєму потоці.

Список літератури

1. Барсегян А.А. Методи і моделі аналізу даних: OLAP і Data Mining : учебное пособие. – Санкт-Петербург: БХВ-Петербург, 2004. – 336 с.

2. Параллельные вычисления CUDA [Електрон. ресурс]. – Режим доступу: http://www.nvidia.ru/object/cuda-parallel-computing-ru.html

3. Big Data – Are You In Control? [Електрон. ресурс]. – Режим доступу: https://www.waterfordtechnologies.com/big-data-interesting-facts/

4. CUDAfy.NET [Електрон. ресурс]. – Режим доступу: https://cudafy.codeplex.com/

УДК 519.68; 681.513.7; 612.8.001.57; 007.51/.52

КУПЦОВА І.В.
ГАВРИЛЕНКО О.В.

МОДИФІКАЦІЯ АЛГОРИТМІВ КЛАСТЕРИЗАЦІЇ ЗА ДОПОМОГОЮ ОБЧИСЛЕННЯ МЕДОЇДІВ КЛАСТЕРІВ ТА ВИКОРИСТАННЯ МЕТРИКИ ХЕМІНГА

В даній статті описано та практично застосовано модифікований метод кластеризації для категоріальних даних. Описуються проблеми класичних алгоритмів, запропоновано метод обробки вхідної вибірки, ініціалізації початкових центрів кластерів, а також алгоритм перевизначення кластерів. На прикладі застосування модифікованого методу до різнопланових вибірок як у якості алгоритму кластеризації, так і алгоритму класифікації була продемонстрована його ефективність.

The subject of the article is development and application of the modified clustering method to categorical data. Problems of classical algorithms are described, method of processing the input sample, defining of the initial centres of clusters and cluster redefining algorithm are proposed. Applying the modified method to different sets of data, both the clustering algorithm and the classification algorithm, its effectiveness was demonstrated.

1. Вступ

Задача чіткої кластеризації полягає в наступному. Нехай маємо вибірку $X = \{x_1, ..., x_n\}$ та функцію відстаней між об'єктами $d(x_i, x_j)$. Необхідно розбити вибірку на множини, які не перетинаються (кластери) так, щоб кожен кластер складався з об'єктів, близьких за метрикою d, а об'єкти різних кластерів суттєво відрізнялись. При цьому, кожному об'єкту x_i приписується мітка кластера $c_i \in C$ [1].

Кластеризація (навчання без вчителя) відрізняється від задачі класифікації (навчання з вчителем) тим, що мітки початкових об'єктів c_i початково не задані, а також може бути невідома сама множина C.

Цілі кластеризації можуть відрізнятися в залежності від особливостей конкретної прикладної задачі:

- зрозуміти структуру множини об'єктів X, розбивши її на групи схожих об'єктів. Спростити подальшу обробку даних та прийняття рішень, працюючи з кожним кластером окремо (стратегія "розділяй та володарюй");
- скоротити об'єм даних, що зберігаються у випадку надвеликої вибірки X, залишивши по одному найбільш типовому представнику від кожного кластера;
- виділити нетипові об'єкти, які не підходять ні одному з кластерів. Цю задачу називають однокласовою класифікацією, визначенням нетиповості чи новизни.

В першому випадку число кластерів намагаються зменшити. В другому – важливіше забезпечити високу степінь схожості об'єктів всередині кожного кластера, а самих кластерів може бути довільна кількість. В третьому – найбільший інтерес представляють окремі об'єкти, які не вписуються ні в один кластер.

2. Проблеми класичних алгоритмів

В класичних алгоритмах машинного навчання існує ряд проблем, пов'язаних з якістю вибірки та деякими додатковими обчисленнями. Зокрема, алгоритм к-середніх стикається з наступними проблемами:

1) розрідженість вибірки, велика кількість унікальних значень;
2) наявність категоріальних ознак, тобто таких ознак, значення яких означає приналежність об'єкта до певної категорії. Безпосередньо самі подібні значення для алгоритмів аналізу даних марні: на практиці різні категорії кодують різними числами, але використовувати на таких даних класичні методи

машинного навчання не можна. Класичний алгоритм k-середніх, також не застосовується для роботи з подібними вибірками;
3) невідома кількість кластерів, в які треба згрупувати дані;
4) залежність результату роботи алгоритму від ініціалізації початкових центрів кластерів;
5) чутливість до появи нових значень.

3. Означення

Об'єкти – елементарна група даних, з якими оперує алгоритм кластеризації. Кожному об'єкту відповідає вектор характеристик [2]:

$$x_i = \left\{ x_{i_1}, ..., x_{i_a} \right\}.$$

Кількість характеристик визначають розмірність простору характеристик.

Відстань – $d(x_i, x_j)$ між об'єктами x_i та x_j – результат застосування обраної метрики.

Медоїд – об'єкт множини даних або кластера, для якого середня відстань до інших об'єктів мінімальна (тобто найближча до центру кластера точка) [3]:

$$x^m = \arg \min_{X \in \{x_1, ..., x_n\}} \sum_{i=1}^{n} d(X, x_i)$$

Ентропія $H(X)$ – міра невизначеності множини даних X:

$$H(X) = \sum_{k \in K} -p(k) \log_2 p(k),$$

де X – поточна множина класів;

K – набір класів в X;

$p(k)$ – пропорція кількості елементів, які потрапили до класу K, до кількості елементів в множині X.

Кількість інформації $IG(a, X)$ – міра різниці ентропії до та після розбиття множини X за атрибутом a:

$$IG(a, X) = H(X) - \sum_{t \in T} p(t) H(t),$$

де $H(X)$ – ентропія множини X;

T – підмножини, утворені шляхом розщеплення множини X за атрибутом a такі, що

$$X = \bigcup_{t \in T} t;$$

$p(t)$ – відношення кількості елементів, що містяться в t до кількості елементів в X;

$H(t)$ – ентропія підмножини t.

4. Математична постановка

Постановка задачі кластеризації пов'язана з визначенням метричного простору

$$(X, d),$$

де $d: X \times X \to R$ – метрика, X – множина об'єктів кластеризації така, що:

$$X = \left\{ X_i \right\}, i = \overline{1, k}.$$

Кластеризація об'єктів множини X представляє собою відображення вигляду:

$$f: X_i \to c_i, i = \overline{1, k},$$

де $C = \left\{ c_i \right\}, k$ – кількість кластерів.

Для кожного кластера c_i з множиною елементів $X_i \subseteq X$ можна визначити медоїд, тобто такий елемент x_i^m, для якого виконується умова:

$$\varphi_x(c_i) = \sum_{x \in X} d(x_{ij}, x_i^m),$$

$$1 \le i \le k, 1 \le j \le p_i,$$

де $d(x_{ij}, x_i^m)$ – відстань від точки x_{ij} до медоїда x_i^m, p_i – потужність множини X_i.

При цьому повинна виконуватись умова:

$$0 \le d(x, x_i^m) \le 1.$$

Необхідно знайти такі $x_i^m, i = \overline{1, k}$, що

$$\sum_{i=1}^{k} d(x_{ij}, x_i^m) \to \min,$$

$$1 \le i \le k, 1 \le j \le p_i.$$

5. Задачі, що необхідно вирішити

Проаналізувавши існуючі методи кластеризації, були виділені наступні задачі, що необхідно вирішити:

1) попереднє опрацювання вибірки;
2) вибір метрики для роботи з категоріальними даними;
3) визначення початкової кількості кластерів;
4) нормалізація даних;
5) ініціалізація початкових медоїд;
6) визначення кластерів.

6. Попередня обробка вибірки

6.1. Фільтрування параметрів вибірки

Задля зменшення часу обробки даних, а також підвищення точності класифікації, вибірка підлягає модифікації до вигляду, який дозволить вилучити та проігнорувати слабковпливові параметри.

В даному методі пропонується вилучати параметри, що задовольняють наступному виразу:

$$IG(a, X) = H(X) \pm \Delta ,$$

де оптимальне значення $\Delta = 0.0127$.

Тобто видаляються параметри, що наповнені різними даними та слабко впливають на результат надання рекомендацій.

6.2. Нормалізація даних

При аналізі даних, їх попередня обробка відіграє велику роль. Нормалізація даних важлива при роботі з параметрами різних одиниць та шкал та використанні метрик визначення відстані.

Тому, для роботи з вхідною вибіркою необхідно провести нормалізацію даних, для чого було обрано метод лінійної нормалізації даних в межах [0,1]:

$$x_i^{'} = \frac{x_i - \min(x)}{\max(x) - \min(x)},$$

де $x = (x_1, ..., x_n)$, x_i – змінна до нормалізації, $x_i^{'}$ – змінна після нормалізації [4].

7. Метрики визначення відстаней

Вибірки з категоріальними даними є проблемою для багатьох алгоритмів машин-ного навчання: для роботи алгоритму, для кожної пари об'єктів необхідно виміряти відстань між ними – степінь схожості.

Класичні метрики, такі як відстань Мінковського, Манхетенська відстань, Евклідова відстань [5,6] недоречні для використання у подібній задачі, так як некоретно відображають степінь близькості об'єктів. Для вирішення цієї проблеми пропонується використовувати метрику Хемінга [7].

Відстань між двома об'єктами x_i та x_j, які містять l категоріальних атрибутів визначається так:

$$d(x_i, x_j) = \sum_{a=1}^{l} \delta(x_{i_a}, x_{j_a});$$

$$\delta(x_i, x_j) = \begin{cases} 0, x_{i_a} \neq x_{j_a} \\ 1, x_{i_a} = x_{j_a} \end{cases},$$

де x_{i_a}, x_{j_a} – значення атрибута (категорії) a в об'єктах x_i та x_j відповідно.

Чим більша кількість невідповідностей категоріальних значень між x_i та x_j, тим більше відрізняються два об'єкти.

8. Визначення початкової кількості кластерів

Для визначення оптимальної кількості кластерів для розбиття вхідної вибірки використовується метод Elbow [8].

9. Ініціалізація початкових медоїдів

Ідея алгоритму ініціалізації початкових медоїдів полягає в тому, щоб обрати нові об'єкти, які знаходяться якнайдалі від існуючих медоїдів.

Алгоритм ініціалізації початкових точок:

1) Обрати перший медоїд x_1^m випадковим чином з X;
2) для кожного $j: j = 1, ..., k$ виконати:

2.1) для всіх x_i:

2.1.1)якщо $j = 1$:обрахувати

$$d_i = d(x_i, x_1^m);$$

2.1.2) інакше: обрахувати

$$d_i = \min(d(x_i, x_1^m), ..., d(x_i, x_j^m));$$

2.2) обрахувати $d_c = \sum_{i=1}^{n} d_i$;

2.3) обрахувати $p_i = \frac{d_i}{d_c}$;

2.4) обрати такий наступний медоїд $x_j^m = x_i : \max(p_i)$.

Чим більша ймовірність p_i, тим далі знаходиться i-й елемент від усіх існуючих медоїдів, що зменшує кількість ітерацій для перевизначення кластерів та допоможе уникнути проблеми виникнення локальних мінімумів при повністю випадковому виборі початкових медоїдів, як пропонується в класичних методах.

10. Визначення кластерів

Задля розподілення об'єктів в кластери, необхідно скористатися метрикою Хемінга, описаною в п.7.

Алгоритм розподілення об'єктів в кластери:

для всіх x_i:

приписати

$$x_i \in c_j \Leftrightarrow j = \arg\min_k d(x_i, x_k^m)$$

11. Алгоритм

Запропонований алгоритм кластеризації категоріальних об'єктів полягає в наступному:

1) Очистити вхідну вибірку (описано в п.6.1);
2) ініціалізувати початкові медоїди (описано в п. 9);
3) визначити кластери (описано в п.10);
4) для кожного кластера c_i:
   - 4.1) визначити випадкову точку x_i в кластері $c_j : x_i \neq c_j$ як новий медоїд $x_i^{'m}$;
   - 4.2) перевизначити кластери відповідно до нового медоїда (описано в п.10);
   - 4.3) якщо $\sum_{i=1}^{n}\sum_{j=1}^{p_i} d(x_{ij}, x_i^m) < \sum_{i=1}^{n}\sum_{j=1}^{p_i} d(x_{ij}, x_i^{'m})$ – повернути останній медоїд та перевизначити кластери відповідно до поверненого медоїда (описано в п.10);
   - 4.4) якщо результат не покращувався вже h кроків – закінчити розгляд поточного кластера c_i.

Параметр h задається вручну, експериментальним шляхом визначено, що при значенні $h = \lfloor 2,6k \rfloor$ досягається оптимальний результат.

12. Визначення якості розбиття

Задля оцінки результатів кластеризації пропонується використовувати метод спрощеного індекса оцінки силуета (Simplified Silhoutte index) [9]

Силует кожного кластера визначається наступним чином: припустимо, об'єкт x_i належить кластеру c_r. Визначимо відстань від цього об'єкта до медоїду того ж самого кластера c_r через a_{ri}. Тепер визначимо середню відстань від x_i до медоїдів інших кластерів $c_t, t \neq r$ через d_{ti}. Покладемо $b_{ri} = \min_{t \neq r} d_{ti}$.

Сенс цієї величини можна визначити як міру несхожості окремого елемента з елементами найближчого кластера. Таким чином, силует кожного окремого елемента визначається як

$$S_{xi} = \frac{b_{ri} - a_{ri}}{\max(a_{ri}, b_{ri})}.$$

Знаменник введено задля нормалізації значень. Очевидно, що високе значення показника S_{xi} характеризує собою кращу приналежність елемента x_i кластеру c_r. Оцінка для всієї кластерної структури досягається усередненням показника по всім елементам:

$$SWC = \frac{1}{n}\sum_{i=1}^{n} S_{xi}.$$

Найкраще розбиття характеризується максимальним SWC, що досягається коли відстань всередині кластера a_{ri} мала, а відстань між елементами сусідніх кластерів b_{ri} велика.

13. Результати

Для випробування запропонованого методу в якості алгоритму кластеризації було згенеровано набір даних, що складається з 3200 об'єктів та 7 категоріальних характеристик.

Для визначення кількості кластерів було використано метод, описаний в п.8.

Табл. 1. Результати кластеризації

Метод	**SWC**
к-середніх	0.812
модифікований метод	0.881

Для випробування модифікованого методу в якості алгоритму класифікації було використано набір даних University Data Set [10], що складається з 2850 записів та 17 категоріальних атрибутів.

Результати класифікації наведені в таблиці 2, де P – показник ефективності, що визначається як відношення правильно класифікованих об'єктів до загальної кількості об'єктів тестової вибірки.

Табл. 2. Результати класифікації

Метод	P
к-середніх	0.887
запропонований метод	0.932

14. Висновок

В статті було виділено основні особливості алгоритмів кластеризації, а також було запропоновано новий метод кластеризації даних. Як виявлено при проведенні експериментів, він дозволяє проводити кластеризацію даних з кращими показниками якості. Варто зазначити, що він може бути використаний також для класифікації даних як алгоритм типу навчання без вчителя. В другому випадку, за рахунок відсутності аналізу попередньої навчальної вибірки, він може показувати гірші результати за алгоритми класифікації типу навчання з вчителем.

Таким чином, розроблений метод дозволяє покращити якість кластеризації у порівнянні з методом к-середніх. В результаті розробки нового методу було вирішено такі задачі:

- підвищення якості вхідної вибірки шляхом її фільтрації;
- уникнення випадкової ініціалізація медоїдів, що призводить до виникнення локальних оптимумів;
- уникнення повного перебору об’єктів при перерахунку медоїдів;
- підвищення стійкості до шумів та аномальних значень.

Список літератури

1. Воронцов К.В. Лекции по алгоритмам кластеризации и многомерного шкалирования.– 2007.– с.2-17
2. Котов А., Красильников Н., Кластеризация данных.–2006.–с.2-16/
3. Машинное обучение [Електронний ресурс] //Режим доступу: http://www.liquisearch.com/what_is_medoid
4. Sandro Saitta. Standartization vs normalization [Електронний ресурс]//.– 2007
5. Amel’kin, S.A., Zakharov, A.V., and Khachumov, V.M., Euclide-Mehalanobis Generalized Distance and Its Properties, *Inform. Tekhnol. Vych. Sist.* .– 2006.– № 4.– с. 40–44
6. Sloane N.J.A., Hardin R.H., Duff T.S., and Conway, J.H., Minimal-Energy Clusters of Hard Spheres, *Discrete Comp. Geom*.– 1995.–№14.–с.237–259
7. Part 2: Hamming Distance//MATH32031: Coding Theory.–с.5-9
8. David J. Ketchen, Jr; Christopher L. Shook. The application of cluster analysis in Strategic Management Research: An analysis and critique//Strategic Management Journal..–1996.–№**17** (6).– с.441–458.
9. Сивоголовко Е.В. Методы оценки качества четкой кластеризации//Компьютерные системы в образовании.–2011.–№4.–с.14-30
10. University DataSet [Електронний ресурс]//Режим доступу: http://archive.ics.uci.edu/ml/machine-learning-databases/university/

УДК 004.93

ЛИТВИН А.В.,
ЛЯШКО С.І.

ПРОСТОРОВА ІНТЕРПОЛЯЦІЯ НА ОСНОВІ ПІДХОДУ ЛОКАЛЬНОГО СУСІДСТВА

У цій статті описані основні методи просторової інтерполяції, зокрема інтерполяції основаної на підході локального сусідства. Очевидно, що за останні десятиліття відбувся суттєвий розвиток у напряму методів просторової інтерполяції, збільшилися обчислювальні потужності систем і тому стало простіше проводити більш точну та стійку просторову інтерполяцію поверхонь. Але все ж нема універсального методу для всіх випадків і потрібно проводити аналіз сфери застосування перед власне застосуванням алгоритму. Адже для вдалої побудови просторової інтерполяції необхідно мати кілька методів інтерполяції для того, щоб вибрати найбільш підходящий для конкретної задачі. У статті наведені основні принципи, переваги та недоліки методів просторової інтерполяції, які основані на підході локального сусідства.

This article describes the basic methods of spatial interpolation, in particular interpolation based on the local neighborhood approach. Obviously, over the past decades, there has been a significant development in the direction of spatial interpolation methods, the computing power of the systems has increased and it is easier to conduct a more accurate and stable spatial interpolation of surfaces. But nevertheless there is no universal method for all cases and it is necessary to carry out the analysis of the sphere of application before the actual application of the algorithm. After all, for successful construction of spatial interpolation it is necessary to have several methods of interpolation in order to choose the most suitable for a particular problem. The article describes the main principles, advantages and disadvantages of spatial interpolation methods based on the local neighborhood approach.

1. Вступ

Просторові та просторово-часові розподіли обох фізичних та соціально-економічних явищ можуть бути апроксимовані функціями залежно від місцезнаходження в багатовимірному просторі, як багатовимірні скалярні, векторні або тензорні поля. Типові приклади - це висоти рельєфу, кліматичні явища, властивості ґрунту, щільність популяції і т. д. Поки більшість із цих явищ характеризуються вимірюваними або оцифрованими точковими даними, які часто нерівномірно розподілені в просторі та часі, візуалізація, аналіз, і моделювання цих явищ зазвичай базуються на растровому представленні. Більше того, явища можна вимірювати за допомогою різних методів (дистанційне зондування тощо), що веде до різнорідних наборів даних з різними цифровими зображеннями та роздільностями, які необхідно об'єднати, щоб створити єдину просторову модель досліджуваного явища.

Багато методів інтерполяції та наближення були розроблені для прогнозування значень просторових явищ для місць, де не проводились вимірювання. В більшості ці методи були розроблені для підтримки перетворення між різними дискретними та безперервними уявленнями просторового та просторово-часового поля, як правило, для перетворення точки або вектора даних у растрове представлення, або для перебудови карти з різною растровою роздільною здатністю.

2. Опис проблеми та критерії її розв'язання

Дано N значень досліджуваного явища $z_j, \quad j = 1, \dots, N$ виміряних значень в дискретних точках $r_j = \left(x_j^{[1]}, x_j^{[2]}, \dots, x_j^{[d]}\right), j = 1, \dots, N$ в межах певної області d-вимірного простору, знайдемо d-вимірну функцію $F(r)$ яка проходить через дані точки, що означає, задовольняє умові:

$$F(r_j) = z_j, j = 1, \dots, N$$

Через те, що існує нескінченна кількість функцій, які виконують цю умову, потрібно накласти додаткові умови, що визначають характер різних методів інтерполяції. Типовими прикладами можуть

бути умови, що засновані на геостатистичних концепціях(Крігінг), місцевість(найближчий сусід), гладкість(сплайни) або спеціально функціональні форми(поліноми). Вибір додаткового стану залежить від характеру модельованого явища та типу застосування.

Пошук відповідних методів інтерполяції викликають кілька проблеми. Модельовані поля зазвичай дуже складні, дані є просторово-гетерогенні і часто є далеко неідеальними та може бути значний шум або розриви. Крім того, набори даних можуть бути дуже великими, які надходять з різних джерел і з різною точністю. Надійні інструменти інтерполяції повинні перш за все задовольняти декільком вимогам: точність та прогностична сила, гнучкість у описі різних типів явищ, згладжування для шумних даних, d-вимірне формулювання, пряма оцінка похідних(градієнти), придатність до великих наборів даних, обчислювальна ефективність та зручність використання.

На даний час важко знайти метод, який виконує всі вищезгадані вимоги для різноманітних типів географічних даних. Тому вибір адекватного методу з відповідними параметрами для конкретного застосування має вирішальне значення. Різні методи можуть давати зовсім різні просторові уявлення, тому потрібне глибоке знання про досліджуване явище, для того, щоб оцінити, який з методів найближчий до реальності. Використання непідходящого методу або невідповідних параметрів може призвести до викривленої просторової моделі, що може призвести до потенційно неправильних рішень. Невірна інтерполяція може мати ще більш глибокий вплив, якщо результат використовуються як вхідні дані для моделювання, де невелика помилка або спотворення може призвести до побудови неправильних просторових візерунків.

3. Методи просторової інтерполяції

Існує дві основні групи методів інтерполяції: детерміновані та геостатистичні. Методи детермінованої інтерполяції створюються поверхні із виміряних точок, базуючись або ступеню подібності, або на рівні згладжування. Геостатистичні методи інтерполяції використовують статистичні властивості виміряних точок. Геостатистичні методи виміряють просторову автокореляцію у виміряних точках і розраховують просторову конфігурацію опорних точок навколо інтерполяційного місцезнаходження.

Детерміновані методи інтерполяції можна розділити на дві групи: глобальні та локальні. Глобальні методи обраховують про інтерпольовані значення на основі всього набору даних. Локальні методи обраховують про інтерпольовані значення на основі виміряних точок в межах околиць: які являються меншими просторовими областями всередині більшої території: що вивчається[1].

Детермінована інтерполяція може змушувати результуючу поверхню проходити через значення даних, або не проходити взагалі. Метод інтерполяції, який вичислює значення ідентичне виміряному в опорному місцезнаходженні, називається жорстким інтерполятором. Нежорсткий інтерполятор обраховує значення, яке відрізняється від виміряного. Останній можна використовувати з метою уникнути виникнення гострих вершин чи заглиблень на висхідній поверхні.

Розглянемо методи детермінованої інтерполяції, яка основана на підході локального сусідства.

4. Підхід локального сусідства

Локальні методи базуються на припущенні, що кожна точка має вплив на результуючу поверхню тільки до повної скінченної відстані. Значення в різних точка обраховуються функціями з різними параметрами, а стан неперервності між цими функціями визначається лише для деяких підходів. Метод вибору точок, який використовується для обчислення інтерполяційної функції, відрізняється між різними методами та їх конкретним способом реалізації.

5. Інтерполяція методом зворотних зважених відстаней

Це один з найпростіших та найбільш доступних методів. Він базується на припущенні, що значення в точці можна апроксимувати як середнєарифметичне значень у точках на певній відстані або від заданого числа m найближчих точок(зазвичай від 19 до 30). Ваги, як правило, обернено

пропорційні до потужності відстані, яка для шуканої локації призводить до оцінки:

$$F(r) = \sum_{i=1}^{m} w_i z(r_i)$$

$$F(r) = \frac{\sum_{i=1}^{m} z(r_j)/|r - r_i|^p}{\sum_{j=1}^{m} 1/|r - r_j|^p}$$

де p - це параметр(зазвичай p=2).

На рисунках 1-2 зображений процес побудови просторової інтерполяції для дискретних точок за допомогою методу зворотних зваженим відстаней у програмній реалізації на Python.

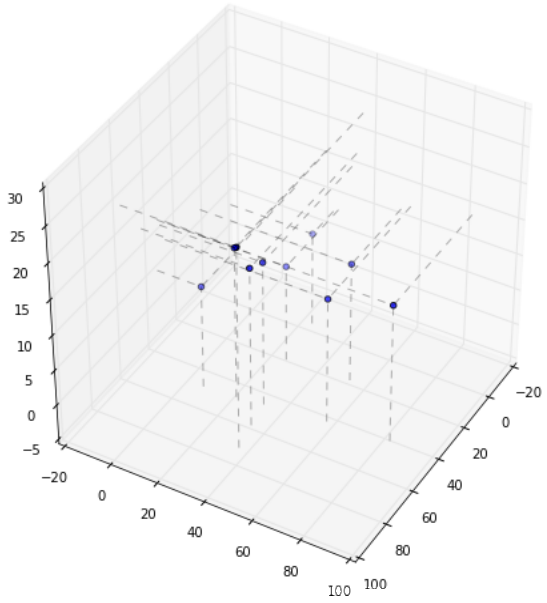


Рис.1. Зображення відомих точок у трьохвимірному просторі

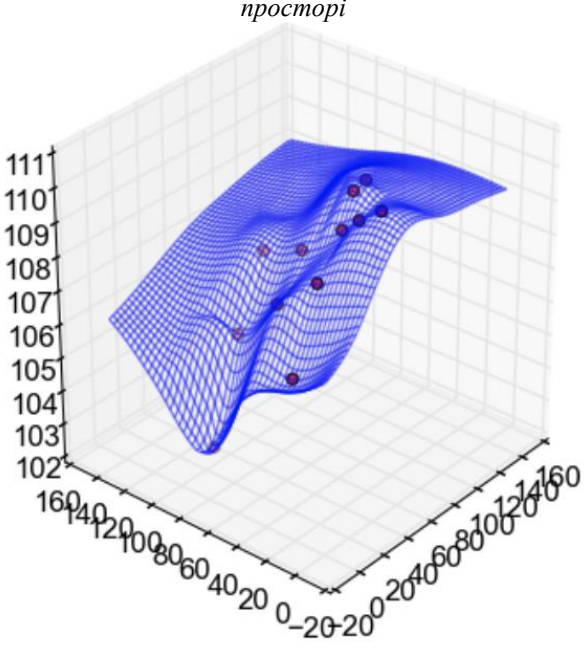


Рис. 2. Інтерпольована поверхня

Хоча цей базовий метод легко реалізувати і він доступний практично в будь-якому ГІС, він має деякі відомі недоліки, які обмежують його практичне застосування[2]. Метод часто не відтворює локальної форми і будує локальні екстремуми в точках даних. Було запропоновано ряд вдосконалень, що призвело до класу багатовимірних IDW поверхонь.

6. Інтерполяція методом природних сусідів

Цей метод використовує середньозважене значення локальних даних на основі поняття природних сусідніх координат, отриманих на основі полігонів Тиссена для двовимірних випадків, та багатокутників Тиссена для трьохвимірного випадку.

Значення в невиміряній точці обчислюється як середньозважене значення найближчих сусідніх значень з вагами залежно від площ або обсягів, а не відстані. Кількість точок, що використовуються для обчислення кожної досліджуваної невідомої точки є змінною залежно від просторової конфігурації даних.

Природна сусідня лінійна інтерполяція приводить до характеристики гумової форми поверхні. Додавання змішаної градієнтної інформації(отриманої за допомогою “попередньої інтерполяції”) дозволяє зробити поверхню гладкою всюди відповідно до характеру явища, що моделюється.

Значення напруженості контролюється двома емпірично обраними параметрами, які змінюються форму функції змішування[3]. Результатом є поверхня з плавнозмінюючими градієнтами, і яка проходить через точки, які змішані з природньо-сусідськими локальними трендами, з локальним регулюючим нахилом та з можливістю обчислення похідних та інтегралів. Цей метод зазвичай використовується для топографічних, батиметричних, геофізичних та ґрунтових даних.

7. Інтерполяція на основі тріангульованої нерегулярної мережі

Цей метод використовує трикутну тесселицію даних точок для отримання двовимірної функції для кожного трикутника, яка потім використовується для оцінки

значень в місцях без вибірки. Лінійна інтерполяція використовує плоскі грані, які відповідають кожному трикутнику. Нелінійні зміщені функції(наприклад, поліноми) використовують для додаткової умови неперервності першого порядку або як похідні першого та другого порядків, що забезпечують плавне з'єднання трикутних і диференційованість отриманої поверхні.

TIN являється формою векторних цифрових географічних даних, які будуються методом триангуляції набору вершин. Вершини з'єднюються серією ребер і формують мережу трикутників. Існують різноманітні методи інтерполяції для формування цих трикутників, приклад триангуляція Делоне.

Отримана триангуляція задовільняє критерію триангуляції Делоне, у відповідності з яким всередині окружностей, описаних через вершини будь-якого із трикутників мережі, не повинно лежати ні однієї вершини цих трикутників. Якщо критерій Делоне виконується по всьому TIN, то мінімальний кут всіх кутів всіх побудованих трикутників максимізується.

Ребра TIN формують неперервні, неперетинаючі трикутники, які можуть використовуватись для визначення положенні лінійних просторових об'єктів. які грають важливу роль в побудові поверхонь.

Вхідні просторові об'єкти, що використовуються для створення TIN, залишаються на тих же місця, де знаходяться вузли та ребра TIN. Це дозволяє зберегти точність вхідних даних при одночасному моделюванні значень, розташованих між відомими точками.

Моделі TIN не так широкодоступні, як растрові моделі поверхонь, і, як правило, їх побудова і обробка дещо дорожча. Вартість отримання високоякісних вихідних даних може бути достатньо високою, а обробка TIN, через складність їх структури, дещо менш ефективна, ніж обробка растрових даних.

Мережі TIN зазвичай використовуються для моделювання невеликих областей з дуже високою точністю, наприклад інженерних застосуваннях, де їх використання дозволяє проводити обчислення планиметричної площі, площі поверхні та об'єму.

Через локальний характер методи зазвичай бувають швидкими, з легким включенням розривів і структурних особливостей. Відповідна триангуляція, що відноситься до геометрії поверхні, має вирішальне значення. Розширення до d-мірних задач являється більш складним ніж для методів, основаних на дистанції. Хоча TIN забезпечує ефективне представлення поверхонь, корисних для різних застосувань, таких як аналіз динамічної візуалізації та видимості, інтерполяція на основі TIN, особливо найпростіша, відносить до найменш точних методів інтерполяції.

8. Висновки

У даній статті наведений огляд основних методів просторової інтерполяції розсіяних даних, зокрема методів, що основуються на підході локального сусідства. Очевидно, що за останні десятиліття відбувся суттєвий розвиток з точки зору точності, надійності, різноманітності застосувань та масштабів вирішення проблем. Однак висновки, що виклав Берроу(1986), знову актуальні: "Нерозумно кидати свої дані в першу доступну техніку інтерполяції без детального вивчення того, як на результати впливатимуть припущення, властиві цьому методу".

Для хорошої побудови карти необхідно мати декілька методів інтерполяції для того, щоб вибрати найбільш підходящий для кожної задачі.

Список літератури

1. Brown W M, Astley M, Baker T, Mitasova H 1995 GRASS as an integrated GIS and visualization environment for spatiotemporal modeling. In Proceedings AutoCarto 12, Charlotte. ASPRS/ASM: 89–99
2. Franke R 1982b Smooth interpolation of scattered data by local thin plate splines. Computers and Mathematics with Applications 8: 273–81
3. Mitas L, Mitasova H 1997 Multivariate approximation by regularized spline with tension. Urbana, National Center for Supercomputing Applications: 1–5.

УДК 004.021

ЛУЦЕНКО В.О.
ЗАДІРАКА В.К.

МОДИФІКАЦІЯ СТЕГОАЛГОРИТМУ НА БАЗІ ТЕОРЕМИ ПРО ДИСКРЕТНУ ЗГОРТКУ ФУНКЦІЇ

В даній статті розглянуто спосіб модифікації стеганоалгоритму на базі теореми про дискретну згортку функції за для підвищення його стеганостійкості. Було проаналізовано ряд удосконалень та змін, проведено порівняльний аналіз з попередніми варіантами реалізації стеганоалгоритму та зроблені відповідні висновки.

In this article, the way steganography algorithm modifications based on discrete convolution theorem function to enhance its steganography algorithm. A number of improvements and changes were analyzed, a comparative analysis of the previous variants of the implementation of the steganography algorithm was carried out and relevant conclusions were drawn.

Вступ

На сьогодні, використання комп'ютерів, ноутбуків та смартфонів стали для людства звичною справою. Електроні пристрої використовуються для збереження, користування та передачі інформації між ними. Але з розвитком технологій передачі інформацій також розвиваються і різні можливості їх крадіжок особами, які не мають прав на доступу до них. Втрата конфіденційної інформації, в результаті, для великих підприємств може обернутись частковою втратою капіталу, а іноді і зовсім призведе до закриття підприємства. Для уникнення ситуацій втрат даних на даний момент існує багато способів збереження та передачі інформації, одним з яких є стеганографія.

Стегоалгоритм на базі теореми про дискретну згортку функції

В роботі [1] були розглянуті два метода стеганографії та був проведений їх порівняльний аналіз за результатами їх виконання. Обидва алгоритма є спектральними, але базуються на різних теоремах. Так, першій базується на використанні похибок заокруглення алгоритму ШПФ, а другий – на базі теореми про дискретну згортку функції. За результатами тестів другий алгоритм виявився кращим ніж перший. Розглянемо цей алгоритм більш детально.

Розглянемо алгоритм якій базується на використанні теореми про дискретну згортку [2]. Нехай є дискретних сигнали $f(k)$ і $g(k)$, що задані N

Відліками кожний. Їх ДПФ відповідно дорівнюють:

$$F(r) = \sum_{k=0}^{N-1} f(k) W_N^{rk}, G(r) = \sum_{k=0}^{N-1} g(k) W_N^{rk}, r = \overline{0, N-1}.$$

Тоді їх циклічна згортка $z(k)$ має вигляд

$$z(k) = f(k) * g(k) = \sum_{t=0}^{N-1} f(k) * g(k - -t),$$

або за допомогою ДПФ, тому що ДПФ згортки $Z(r)$ є добутком ДПФ $F(r)$ та $G(r), r = \overline{0, N-1}$ [3]:

$$Z(r) = F(r) * G(r). \quad (1)$$

Отже для обчислення згортки за допомогою ДПФ необхідно виконати наступне:

1. для заданих сигналів f(k) та g(k) обчислюються їх ДПФ F(r) та G(r);
2. отримані F(r) та G(r) перемножуються для одних і тих же r;
3. для знайденого таким чином відповідно до (1) спектра згортки Z(r) обчислюється ОДПФ.

Щоб позбутися необхідності кожний раз підбирати порожній контейнер під повідомлення, яке передається, пропонується стеганоалгоритм з практично не змінюваним в процесі обробки контейнером. Будується таке перетворення Q, яке дозволяє наблизити стегоконтейнер — згортку z(k) до порожнього контейнера g(k): Q(f(k)) = h_l(k), де H_1(r) = 1 — ДПФ h_1(k), що забезпечує виконання рівності: f(k) ∗g(k) ≈ g(k).

Також у статі [4] було показано, що за умови, коли для сигналу-повідомлення виконується f(0) ≠ 0, як допоміжний сигнал $h_1(k)$ пропонується функція:

$$h_1(k) = \frac{x(k)*f(k)}{f(0)},$$

де $x(k) = \frac{1}{1+\alpha_1*k^2}, \alpha_1 > 0, k = \overline{0, N-1}.$

Розглянемо сам стеганоалгоритм наведений у [4]:

1. нехай маємо сигнал-повідомлення f(k) та порожній контейнер g(k), $k = \overline{0, N-1}$, параметри якого є ключем K;
2. визначимо допоміжний сигнал $h_1(k)$ = Q(f(k)), $k = \overline{0, N-1}$;
3. обчислимо спектри $H_1(r)$ та G(r);
4. отримаємо оцінку спектра їх згортки $Z_1(r) = H_1(r)*G(r)$;
5. виконаємо ОДПФ $Z_1(r)$ і отримуємо згортку $z_1(k) = h_1(k)*g(k)$, $k = \overline{0, N-1}$, близьку до порожнього контейнера g(k);
6. пересилаємо стегоконтейнер-згортку по відкритому каналу зв'язку, а ключову інформацію — по закритому;
7. одержувач обчислює ДПФ заповненого контейнера $Z_1(r)$ та ДПФ відновленого за ключем порожнього контейнера G(r);
8. ділить $Z_1(r)$ на G(r) для тих самих r, обчислюючи таким чином спектр допоміжного сигналу $H_1(r)$;
9. обчислює ОДПФ $H_1(r)$, отримуючи $h_1(k)$;
10. виконуючи обернене перетворення $Q^{-1}(h_1(k))$ = f(k), отримує вихідний сигнал-повідомлення.

Розглядаючи перетворення Q(f(k)) в роботі [4] були неведені наступні функції:

- функція Гаусса: $f_1(x) = e^{-ax^2}$;
- функція Лапласа (експонента за модулем): $f_2(x) = e^{-a|x|}$;
- функція Лоренца: $f_3(x) = \frac{1}{1+ax^2}, a > 0, a \to +\infty$.

Використовуючи ці функції, побудували перетворення Q:

$$Q_1\big(f(k)\big) = h_1(k) = \frac{f(k)}{f(0)}\sqrt{N}f_1(k), f(0) \neq 0, k = \overline{0, N-1} \quad (2)$$

$$Q_2\big(f(k)\big) = h_2(k) = \frac{f(k)}{f(0)}\sqrt{N}f_2(k), f(0) \neq 0, k = \overline{0, N-1} \quad (3)$$

$$Q_3\big(f(k)\big) = h_3(k) = \frac{f(k)}{f(0)}\sqrt{N}f_3(k), f(0) \neq 0, k = \overline{0, N-1} \quad (4)$$

Очевидно, що коефіцієнти ДПФ отриманих функцій будуть наближатися до одиничних значень при збільшенні параметра a. Але навіть при малих значеннях a функції (2) та (3) швидко прямують до нульових значень, що призводить до втрати даних. Провівши експерименти над (4) функцією булу отримано, що a може змінюватися від 0 до 10^7, при стандартній точності обчислень обраної мови програмування – JavaScript стандарту 2017 року [5] в якому використовуються числа з плаваючою комою які реалізовані за стандартом IEEE 754-2008 [6] типу binary64 з 16 значущими знаками після коми. Але цей проміжок зменшується через те, що при доволі малих значеннях a порожній контейнер доволі сильно спотворюється при вкраплені таємної інформації, що поліпшує її знаходження та вилучення. Тому насправді ми можем використовувати a у проміжку від 10^4 до 10^7, що вже є прекрасним результатом але і в такому випадку можна перебрати всі можливі варіанти, та отримати секретне повідомлення. Тому було прийнято рішення замінити функцію на іншу з більшим проміжком можливих значень її коефіцієнтів для збільшення стеганостійкості алгоритму.

Розглянемо наступні функції:

$$f_4(x) = \frac{1}{1+a*|\sin(b*x)|},$$

$$f_5(x) = \frac{1}{1+a*|\sin(b*x)|+c*x^2}.$$

Використовуючи ці функції, побудували перетворення Q:

$$Q_4\big(f(k)\big) = h_4(k) = \frac{f(k)}{f(0)}\sqrt{N}f_4(k), f(0) \neq 0, k = \overline{0, N-1}, \quad (5)$$

$$Q_5\big(f(k)\big) = h_5(k) = \frac{f(k)}{f(0)}\sqrt{N}f_5(k), f(0) \neq 0, k = \overline{0, N-1}. \quad (6)$$

Також коефіцієнти ДПФ запропонованих функцій будуть наближатися до одиничних значень при збільшенні їх параметрів a, b і c (рис. 1).

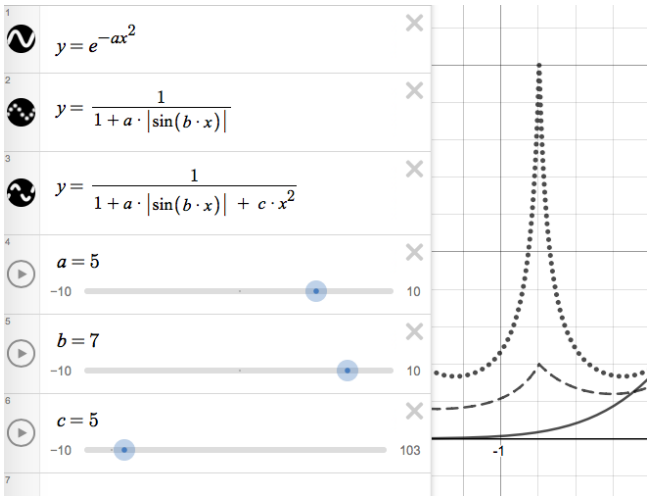


Рис. 1. Графіки функцій f3, f4, f5 при a=5, b=7, c=5

$$\widehat{Q_4}(r) = \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} Q_4\left(f(k)\right) W_N^{rk}$$

$$= \sum_{k=0}^{N-1} \frac{f(k)}{f(0)} f_4(k)\, e^{-\left(ak^2 + \frac{2\pi i}{N} rk\right)} =$$

$$= 1 + \frac{f(1)}{f(0)} f_4(1) e^{-\left(a + \frac{2\pi i}{N} r\right)} + \cdots + \frac{f(N-1)}{f(0)} f_4(N - 1) e^{-\left(a(N-1)^2 + \frac{2\pi i}{N} r(N-1)\right)},$$

$$r = \overline{0, N-1}$$

$$\widehat{Q_5}(r) = \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} Q_5\left(f(k)\right) W_N^{rk}$$

$$= \sum_{k=0}^{N-1} \frac{f(k)}{f(0)} f_5(k)\, e^{-\left(ak^2 + \frac{2\pi i}{N} rk\right)} =$$

$$= 1 + \frac{f(1)}{f(0)} f_5(1) e^{-\left(a + \frac{2\pi i}{N} r\right)} + \cdots + \frac{f(N-1)}{f(0)} f_5(N - 1) e^{-\left(a(N-1)^2 + \frac{2\pi i}{N} r(N-1)\right)},$$

$$r = \overline{0, N-1}$$

Оскільки за побудовою $0 \le f(k) \le 2^m$, то при $a \to \infty$ маємо $\widehat{Q_4}(r) \approx 1$, $\widehat{Q_5}(r) \approx 1$ для $r = \overline{0, N-1}$.

За припущенням $f_4(k) \neq 0$ та $f_5(k) \neq 0$, запропоновані перетворення – обернені. Але при практичній реалізації алгоритму ми переходимо від реальних значень до їх наближень з деякою наперед заданою точністю.

Серія чисельних експериментів показала, що мінімальні значення коефіцієнтів для перетворення Q_4 (5) за якими досягається мінімально допустимі спотворення порожнього контейнера дорівнюють $a=10^7$, b=1. Оцінка похибки, привнесеної в результаті вкраплення повідомлення в порожній контейнер за першою нормою $\varepsilon_1 = \max\limits_{k=\overline{0,N-1}} |g(k) - z_1(k)| = 8.76 * 10^{-7}$, за евклідовою нормою $\varepsilon_E = \frac{\sqrt{\sum_{k=0}^{N-1} \left(g(k) - z_1(k)\right)^2}}{N} = 1{,}23 * 10^{-7}$. Для Q_5 (6) за якими досягається мінімально допустимі спотворення порожнього контейнера дорівнюють $a=10^6$, b=1, c=1. Оцінка похибки, привнесеної в результаті вкраплення повідомлення в порожній контейнер за першою нормою $\varepsilon_1 = 7.82 * 10^{-7}$, за евклідовою нормою $\varepsilon_E = 1{,}36 * 10^{-7}$. Експерименти проводилися для N=8 та N=16. Також було знайдено максимальне значення коефіцієнтів при якому виходить вилучити повідомлення із стегоконтейнера без втрат, так які забезпечують надійну стеганостійкість, а саме $a = \overline{0, 10^{10}}$, $b = \overline{0, 10^{15}}$ та $c = \overline{0, 10^{15}}$, що дає значно більшу кількість варіантів побудови таємного ключа для вкраплення та вилучення ніж при використанні функції Лоренца.

Отже, при практичній реалізації цього алгоритму можливе застосування наведених функцій, в результаті чого можлива побудова стійкої відносно атак пасивного противника стеганографічної системи.

Висновки

В даній статі було розглянуто алгоритм для розв’язання задач комп’ютерної стеганографії, його теоретичне підґрунтя та основні характеристики. Було розглянуто його основні недоліки та запропоновано варіанти їх вирішення. У майбутньому можна буде знайти інші функції з більшою кількістю коефіцієнтів що підвищить стеганостійкість алгоритму.

Список літератури

1. В. К. Задірака, І. В. Сергієнко, О. М. Литвин, С. С. Мельникова и О. П. Нечуйвітер, Оптимальні алгоритми обчислення інтегралів від швидко-осцилюючих функцій та їх застосування. Том 2. Застосування., Київ: Наукова думка, 2011, p. 348.
2. Н. В. Бородавка и В. К. Задирака, «Стеганоалгоритмы на базе теоремы о свертке,» Кибернетика и системный анализ, pp. 139-144, 2004.
3. Л. Робинер и Б. Гоулб, «Теория и применения цифоровой обработки сигналов,» Мир, 1978, p. 848.
4. В. К. Задирака и Н. В. Кошкина, «Спектральные методы решения задач компьютерной стеганографии,» Проблемы управления информатики, т. 4, pp. 132-151, 2011.
5. Ecma Internationa, "Ecma-262," Червень 2017. [Online]. Available: http://www.ecma-international.org/publications/files/ECMA-ST/Ecma-262.pdf.
6. IEEE Computer Society, 29 Серпень 2008. [В Интернете]. Available: http://eng.umb.edu/~cuckov/classes/engin341/Reference/IEEE754.pdf.

УДК 004.94; 537.63

МАЧУЛЯНСЬКИЙ Д.І.
ЛЯШКО С.І.,

ПРАКТИЧНЕ ЗАСТОСУВАННЯ МАТЕМАТИЧНОЇ МОДЕЛІ ВЗАЄМОДІЇ МАГНІТНОГО ПІДВІСУ З РЕАКТОРОМ ДЛЯ ВІЗУАЛІЗАЦІЇ МАГНІТНОГО ПОЛЯ

В даній статті розглянуто практичне застосування моделі взаємодії магнітного підвісу з реактором для візуалізації магнітного поля. Розроблено математичну модель взаємодії магнітного підвісу з використанням левітуючого диполю у термоядерному реакторі та проведено аналіз його основних параметрів з метою виявлення та їх оцінки, котрі значно впливають на роботу термоядерного реактора. Використання математичної моделі дозволяє змоделювати процес роботи магнітного підвісу та дослідити його за параметрами.

Ключові слова: ВІЗУАЛІЗАЦІЯ, ЛЕВІТУЮЧИЙ ДИПОЛЬ, МАГНІТНЕ ПОЛЕ, ТЕРМОЯДЕРНИЙ РЕАКТОР, МАГНІТНИЙ ПІДВІС.

In this article, we consider the practical application of the model interaction of a magnetic suspension with a reactor for magnetizing field visualization. The mathematical model of the interaction of a magnetic suspension with the use of a dipole in the thermonuclear reactor was developed and an analysis of its main parameters was made to detect and evaluate them, which greatly influence the operation of the thermocuclear reactor. Using mathematical model allows to simulate the process of magnetic suspension and to study it by parameters.

Key words: VISUALIZATION, LEVITATED DIPOLE, MAGNETIC FIELD, FUSION REACTOR, MAGNETIC APPARATUS.

1. Вступ

Дипольне магнітне поле - це найпростіша і найпоширеніша конфігурація магнітного поля всесвіту. Це магнітне дальнє поле єдиного кругового струму, яке представляє домінуючу структуру середніх магнітосфер намагнічених планет і нейтронних зірок. Використання дипольного магнітного поля, створеного левітовим кільцем для обмеження гарячої плазми для генерації синтезу енергії, вперше було розглянуто Акірою Хасегава.

Особливість магнітного поля, яке створюється в реакторі з левітуючим диполем, пов'язана з необхідністю забезпечення магнітної левітації центральної надпровідної котушки (ЦНК).

Таким чином, струм в ЦНК складається з власного струму і струму, що індукується іншими джерелами магнітного поля, які безконтактно взаємодіють з ЦНК. Будь-які коливання ЦНК поблизу положення рівноваги призводять до зміни картини магнітного поля. Магнітне поле ЦНК - це плазмова пастка. Важливість його візуалізації для роботи реактора є очевидною. Зазвичай для забезпечення стійкості безконтактної рівноваги левітуючого диполю використовується система автоматичного регулювання [1], яка керується мікропроцесорами. Ми пропонуємо використовувати альтернативний механізм стабілізації ЦНК, який заснований на фізичних принципах [2].

Для дослідження цього механізму з метою застосування для левітуючого диполю необхідна візуалізація магнітного поля, як повної системи, так і її частин. Також необхідно відображення "силових ліній" магнітного поля в статиці та в динаміці. Необхідне дослідження коливань поля в області формування та утримання плазмового згустку, що потрібно для відповіді на питання про можливість застосування нового підходу з левітуючим диполем.

2. Виклад основного матеріалу

Для візуалізації магнітного поля та дослідження взаємодії магнітного підвісу з реактором необхідно проаналізувати залежності «горизонтальної» і «вертикальної» складових вектора магнітної індукції над поверхнею несучого току сфери від точки спостереження.

Закон Біо-Саварра-Лапласа і принцип суперпозиції дозволяють розрахувати індукцію магнітного поля $\vec{B}$, створюваного довільній системою електричних струмів, в довільній точці простору. Для цього необхідно розбити всі струми на нескінченно малі ділянки $(I\Delta\vec{l})_k$, записати вирази для векторів для індукції поля $(\Delta\vec{B})_k$, що створюються цими елементами (користуючись законом Біо-Саварра-Лапласа) і підсумувати отримані вирази (що дозволяє принцип суперпозиції)

для всіх ділянок струму. Отже, застосуємо це для дослідження в нашому прикладі.

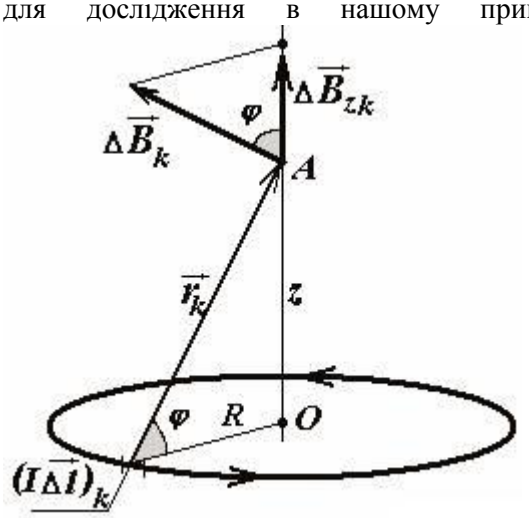


Рис. 1. Схема для знаходження індукції поля в точці А, яка знаходиться на осі кільця на відстані z від його центру.

Точка А з рис. 1 - центр левітуючого диполю, для її дослідження необхідно виділити малу ділянку кільця $(I\Delta\vec{l})_k$ і будуємо вектор індукції поля $(\Delta\vec{B})_k$, створеним цим елементом, в даній точці. Цей вектор перпендикулярний вектору $\vec{r}$, що з'єднує виділену ділянку з точкою спостереження. Вектори $(I\Delta\vec{l})_k$ і $\vec{r_k}$, як і раніше, перпендикулярні, тому $sin\alpha = 1$. Оскільки кільце має осьову симетрію, то сумарний вектор індукції поля в досліджуваній точці повинен бути спрямований по осі кільця. Таким чином, для того щоб визначити модуль сумарного вектора індукції, необхідно підсумувати проекції векторів на вісь кільця. Ця операція не представляє особливої складності, якщо врахувати, відстані від усіх точок кільця до точки спостереження однакові $r = r_k = \sqrt{R^2 + z^2}$, а також однакові кути φ між векторами $(\Delta\vec{B})_k$ і віссю кільця. Запишемо вираз для модуля шуканого сумарного вектора індукції

$$B = \sum_k \Delta B_{zk} = \sum_k \frac{\mu_0}{4\pi}\frac{(I\Delta\vec{l})_k}{r^2}\cos\varphi = \frac{\mu_0}{4\pi}\frac{I\cos\varphi}{r^2}\sum_k(\Delta l)_k = \frac{\mu_0}{4\pi}\frac{I\cos\varphi}{r^2}2\pi R = \frac{\mu_0 IR}{2r^2}\cos\varphi)$$

Використовуючи загальний розглянутий метод, можна розрахувати індукцію поля в довільній точці. Вже згадана система має осьову симетрію, тому досить знайти розподіл поля в площині, перпендикулярній площині кільця і проходить через його центр. Можна показати, що компоненти вектора магнітної індукції поля, створюваного одним виділеним елементом струму, в точці з координатами (y, z) розраховуються за формулою (1).

$$r_k = \sqrt{x^2 + y^2 - 2xR\cos\varphi_k + 1};$$
$$\Delta B_{yk} = -\frac{\mu_0}{4\pi}\frac{z\cos\varphi_k}{r_k^3}\Delta\varphi; \qquad (1)$$
$$\Delta B_{zk} = -\frac{\mu_0}{4\pi}\frac{1-y\cos\varphi_k}{r_k^3}\Delta\varphi.$$

Якщо відомо значення вектора індукції в кожній точці, то можна побудувати картину силових ліній магнітного поля, приклад якої наведено на рис. 2.

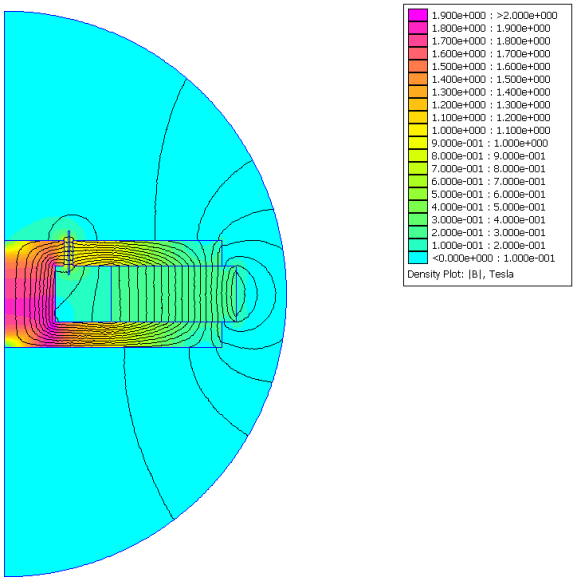


Рис. 2. Щільність потоку магнітних ліній в залежності від площі.

3. Висновок

Енеретична криза є актуальною проблемою для людства. Тому сприяння розвитку знаходження та використання альтернативного палива є значним вкладом в наше майбутнє. Дані результати посприяють інтенсивному дослідженню та використання левітуючого диполю, особливо для зменшених у габаритах реакторах, які NASA збирається використовувати для своїх космічних кораблів.

Список літератури

1. LDX Progress Report [Електронний ресурс]. – Режим доступу: https://www-internal.psfc.mit.edu/ldx/reports/status_0702.html (останній візит: 14.04.18)
2. Демуцкий В.П., Зуб С.С., Рашкован В.М.
Анализ устойчивости статического равновесия свободной сверхпроводящей катушки в системе трех жестко связанных сверхпроводящих катушек
// Вісник харківського університету. –1999. – вип.2 (6), 443. –С.34-40.
3. Maxwell J.C. On physical line of force. - London. -1861.
4. Козорез В.В. Динамика систем магнитно взаимодействующих тел. - Киев. - 1966.
5. Фихтенгольц Г.М. Курс дифференциального и интергального исчисления. - Москва. - 1966.

УДК 519.2

ПЕРЕРВА А.С.
КОВАЛЮК Т.В.

ОГЛЯД МЕТОДІВ АНАЛІЗУ ФІНАНСОВИХ ЧАСОВИХ РЯДІВ

У даній статті розглянуто методи аналізу часових рядів та принципи їх використання. Проведення аналізу фінансових часових рядів дає змогу здійснити економічний прогноз діяльності підприємства на майбутній період. Теоретичні дослідження, засновані на аналізі часових рядів - потужний інструмент для дослідження багатьох явищ. Саме тому існує чимало методів для аналізу часових рядів. Їх порівняльний аналіз дає розуміння принципів, переваг і недоліків кожного з них.

In this article the methods of analysis of time series and the principles of their usage are considered. The analysis of financial time series allows to make a forecast of the company's activities for the future period. Theoretical researches based on the analysis of time series are a powerful tool for understanding many phenomena. Their comparative analysis gives an understanding of the principles, advantages and disadvantages of each of them.

Ключові слова: часовий ряд, аналіз, метод аналізу, кореляційний аналіз, спектральний аналіз, ковзне середнє, згладжування, прогнозува

1. Вступ

Наразі, для вивчення властивостей складних систем, в тому числі і при експериментальних дослідженнях, широко використовується підхід, заснований на аналізі сигналів, вироблених системою. Даний підхіж є актуальним в тих випадках, коли математично описати досліджуваний процес практично неможливо, але в розпорядженні дослідника є деяка характерна спостережувана величина. Тому аналіз систем, особливо при експериментальних дослідженнях, часто реалізується за допомогою обробки, або ж аналізу, реєстрованих сигналів. Наприклад, в аритмології в якості такого сигналу використовується електрокардіограма, в сейсмології - запис коливань земної кори, в метеорології - дані метеоспостережень, в економіці - фінансові показники за певний період часу і т.п.

Аналіз часового ряду і подальше прогнозування його розвитку може використовуватися для:

1. планування в економіці, виробництві, торгівлі;
2. управління та оптимізації, що протікають в суспільстві соціально-економічних процесів;
3. часткового управління важливими параметрами демографічних процесів та екологічної ніші суспільства;
4. прийняття оптимальних рішень в бізнесі.

Мета: розглянути можливість практичного застосування методів аналізу фінансових часових рядів.

Завдання: Показати потужність теоретичних досліджень на основі аналізу фінансових часових рядів та їх значущість для успішного фінансового розвитку підприємництва. Описати механізм зміни рівнів ряду, що використовується для статистичного прогнозування, яке, як правило, зводиться до екстраполяції виявлених тенденцій розвитку.

2. Виклад основного матеріалу

Часовий ряд - це набір числових даних, отриманий протягом послідовних періодів часу.

Метод аналізу часових рядів дозволяє передбачити значення числової змінної на основі її минулих і теперішніх значень.

Наприклад, щоденні котирування акцій на фондовій біржі утворюють часовий ряд. Іншим прикладом часового ряду є щомісячні значення індексу споживчих цін, щоквартальні величини валового внутрішнього продукту і щорічні доходи від продажів якої-небудь компанії.

Існує велика кількість різних методів аналізу часових рядів. В даній статті розглянуто деякі з них, такі як:

1. Кореляційний аналіз
2. Спектральний аналіз
3. Метод згладжування.

3. Огляд алгоритмів

Кореляційний аналіз

Даний метод аналізу дозволяє виявити суттєві періодичні залежності і їх затримки всередині одного процесу (автокореляція) або між декількома процесами (кроскореляція).

Кореляційний аналіз досить простий у використанні, при цьому він дозволяє визначити вплив різних економічних показників. Тому даний метод аналізу досить популярний у економістів. Класична кореляційна теорія була розроблена для випадкових стаціонарних процесів, що допускають отримання будь-якого числа реалізацій (спостережень, вимірювань). Під стаціонарністю розуміється незмінність середнього рівня випадкового процесу дисперсії, відхилень, і сталості автокореляційної функції. Ця теорія, перш за все, була розроблена для технічних додатків, де гіпотеза про стаціонарність являється прийнятною, а принципових обмежень на число реалізації немає. Саме ця теорія, розроблена для стаціонарних випадкових процесів, не критично сприймається економістами і використовується для аналізу процесів свідомо не стаціонарних (наприклад, що мають тренд) і навіть не випадкових, які зазвичай представлені єдиною реалізацією. Однак велика кількість реалізацій в економіці, при одних і тих же умовах, як правило, неможлива. До того ж вибіркові пари, що беруть участь в кореляційному аналізі, повинні бути незалежними, але і ця умова порушується в тимчасових рядах, так як сполучені ознаки не є незалежними хоча б тому, що пов'язані певними моментами часу, тобто підпорядковані певному хронологічному порядку, і їх неможливо переставити. Постулювання стаціонарності процесів x_1 і x_2 - це вимушений захід, до якої економісти вдаються, щоб забезпечити собі хоч якусь можливість провести кореляційний аналіз зв'язку двох змінних. Насправді змінні x_1 і x_2, як правило, не мають фіксованих середніх рівнів і будь-яких певних середньоквадратичних відхилень від них. У даному випадку звичайні показники кореляції виражають швидше зв'язок між відхиленнями рядів від їх середніх, ніж між самими рядами. Слід зазначити, що в разі попереднього виключення тренду спотворення первинної інформації тільки посилюється. Проте, є інший вихід - судити про наявність позитивної або негативної кореляції за випадковим збігом або не збігом знаків приросту змінних. Для отримання загального уявлення про усередненні кореляційні властивості двох нестаціонарних рядів було розроблено модифікований коефіцієнт кореляції:

$$r_{mod} = \frac{\sum_{t=2}^{n} \Delta y_{1t} \Delta y_{2t}}{\sum_{t=2}^{n} \Delta y`_{1t} \Delta y`_{2t}} \qquad (1)$$

Знаменник формули (1) - норма коефіцієнта. Завдяки ньому r_{mod} не може вийти за межі $-1 \leq r_{mod} \leq 1$. При такому вимірнику ступеня кореляції двох рядів, недоліки усуваються, і ніякої деформації вихідних даних не відбувається. При значеннях $r_{mod} < 0{,}4$ кореляційний зв'язок вважається слабким.

Принципи кореляційного методу аналізу:

1. Кореляційний зв'язок тимчасових рядів не є постійним показником, а змінюється у часі.
2. При аналізі кореляційного зв'язку тимчасових рядів недоцільно використовувати коефіцієнт кореляції збільшень як постійну величину.
3. Коефіцієнт кореляції, розрахований на кожен момент або період часу по накопиченим підсумкам приросту часових рядів, є функцією часу, тобто часовим рядом, який можна використовувати для аналізу кореляційного зв'язку розглянутих в часі явищ.

Спектральний аналіз

Спектральний аналіз дає змогу знаходити періодичні і квазіперіодичні складові часового ряду. Зазвичай цей метод аналізу використовують для вивчення коротких часових рядів довжиною від декількох сотень до декількох тисяч спостережень. В економічних даних таку довжину можуть мати ряди щоденної, щотижневої або щомісячної кон'юнктурної інформації по товарних ринках. В даний час у зв'язку з жвавим інтересом до теорії довгих хвиль часто вдаються до аналізу річних даних.

Довжина ряду визначається потребами задачі. Для управлінської діяльності важливо знати закони, які хоча і виявляються на тлі будь-яких тривалих тенденцій, але їх вплив на економічне життя є визначальним для порівняно короткого тимчасового інтервалу і важливі для короткострокового і середньострокового прогнозування. Ставлячи довжину інтервалу спостережень, ми фактично фільтруємо ряд, виключаючи можливість спостерігати в динаміці циклічності з періодом, що перевищує його довжину. У коротких рядах фрагменти довгих хвиль сприймаються як тренди. Короткими тимчасовими рядами будемо називати ряди, що мають 20-50 спостережень. Проблеми дослідження таких рядів пов'язані, перш за все, з надійністю отриманих оцінок функції автоковаріації і розраховуються на їх основі оцінок функції спектральної щільності. Надійність оцінок періодів коливань падає з ростом періоду. У зв'язку з цим слід усвідомлювати якісну різницю спектральних оцінок в різних частотах. Тривалі залежності не можна вважати надійними, і тому питання про них як закономірностях процесу залишається відкритим. Щоб зробити висновок про те, що виявлені тривалі залежності є закономірностями, внутрішньо притаманними розвитку даного процесу, слід, якщо можливо, перейти до розгляду показника за 100 і більше років, або залучити додаткові якісні уявлення про процес. Середні і короткі циклічності з більшою ймовірністю можна вважати закономірностями розвитку. Ще одна проблема, яка набуває для коротких рядів особливої гостроти - це нестаціонарність процесів. Щоб якомога менше спотворити динаміку ряду, не втратити цінні спостереження і позбутися від тренду середнього, для коротких рядів перед представляється виправданим побудова поліноміальних трендів з метою отримання доступу до відхилень фактичних даних від тренду і вилучення відомостей про приховані в динаміці ряду закономірності. Поступово збільшуючи ступінь полінома, можна контролювати вплив фільтрації, будуючи графік функції спектральної щільності відхилень від тренду. Слід зупинити процес підвищення ступеня полінома відразу, як тільки функція почне "працювати", тобто як тільки сплеск функції спектральної щільності в околиці нульової частоти зменшиться настільки, що стане можливим судити про структуру процесу за оцінками функції в інших частотах. На практиці ступінь полінома для коротких економічних рядів, як правило, не перевищує двох. Включення третього ступеня найчастіше погіршує статистичні характеристики рівняння (коефіцієнт при цьому факторі виходить незначним і збільшується помилка апроксимації.) Для моделювання процесів важливо знати, чи є тренд середнього. У багатьох випадках достатньо аналізу графіка, але іноді можна дати впевнену відповідь. Можна використати статистичний критерій Манна для перевірки наявності тенденції ряду до зростання або зменшенням. Використання критерію Манна дає однозначну відповідь про наявність чи відсутність тренда в даних, але при використанні спектральної техніки розрахунок критерію являється зайвим. Функція спектральної щільності дає вичерпну інформацію про нестаціонарність процесу: про наявність тренду і його ролі в формуванні дисперсії. За співвідношенням оцінки в нульовій частоті і величин сплесків в інших частотах функціях спектральної щільності дозволяє зрозуміти роль в динаміці ряду його складових компонент.

Згладжування

Даний метод призначений для перетворення часових рядів з метою видалення з них високочастотних або сезонних коливань.

Найпростіший метод згладжування рядів — ковзне (плинне) середнє. Його сутність полягає в тому, що для будь-якої непарної кількості точок послідовності ряду замінювати центральну точку на середнє арифметичне решти точок:

$$s_i = \frac{1}{2k+1} \sum_{j=-k}^{k} x_{i+j} \qquad (2)$$

де – x_i x_iпочатковий ряд, s_i– згладжений ряд.

Метод ковзного середнього має певні недоліки:

1. Ковзне середнє неефективно в обчисленні. Для кожної точки середнє необхідно переобчислювати по новому. Не можна перевикористати результат, обчислений для попередньої точки.

2. Ковзне середнє не можна продовжити на перші і останні точки ряду. Це може викликати проблему, якщо нас цікавлять саме ці точки.

3. Ковзне середнє не визначене за межами ряду, і як наслідок, не може використовуватися для прогнозування.

Більш просунутий метод згладжування, який також можна використовувати для прогнозування - експоненціальне згладжування. Іноді його називають методом Хольта-Уінтерс (Holt-Winters) в честь імен його творців.

Існує декілька варіантів даного методу:

1. одинарне згладжування для рядів, у яких немає тренду і сезонності;

2. подвійне згладжування для рядів, у яких є тренд, але немає сезонності;

3. потрійне згладжування для рядів, у яких є і тренд, і сезонність.

Метод експоненціального згладжування обчислює значення згладженого ряду шляхом оновлення значень, розрахованих на попередньому кроці, використовуючи інформацію з поточного кроку. Інформація з попереднього і поточного кроків береться з різними вагами, якими можна управляти.

У найпростішому варіанті одинарного згладжування співвідношення таке:

$$s_i = ax_i + (1 - a)_{S_{i-1}} \qquad (3)$$

де $0 \leq \alpha \leq 1$.

Параметр α визначає співвідношення між незгладженим значенням на поточному кроці і плавним значенням з попереднього кроку. При α = 1 беруться тільки точки вихідного ряду, тобто ніякого згладжування не буде. При α = 0 беруться тільки згладжені значення з попередніх кроків, тобто ряд перетворюється в константу.

Зі співвідношення видно, що всі попередні значення ряду вносять вклад в дане згладжене значення, однак їх внесок згасає експоненціально за рахунок зростання ступеня параметра α.

Однак, якщо в даних є тренд, просте згладжування буде «відставати» від нього (або доведеться брати значення α близьким до 1, але тоді згладжування буде недостатнім). Потрібно використовувати подвійне експоненціальне згладжування.

Подвійне згладжування використовує вже два рівняння - одне рівняння оцінює тренд як різницю між поточним і попереднім згладженим значеннями, потім згладжує тренд простим згладжуванням. Друге рівняння виконує згладжування як у простому варіанті, але в другому додатку використовується сума попереднього згладженого значення і тренду.

Потрійне згладжування включає ще один компонент - сезонність, і використовує ще одне рівняння. При цьому розрізняють два варіанти сезонного компонента - адитивний і мультиплікативний. У першому випадку амплітуда сезонного компонента постійна і з часом не залежить від базової амплітуди ряду. У другому випадку амплітуда змінюється разом зі зміною базової амплітуди ряду. З ростом ряду амплітуда сезонних коливань збільшується.

Кожен рівень тимчасового ряду пов'язаний з відповідним моментом часу або часовим інтервалом. Рівні ряду за своїм змістом мають бути порівнюваними. Показники часових рядів формуються під впливом великої кількості короткодіючих та довготривалих факторів, в тому числі випадковостей різного роду. Зміна умов розвитку явищ супроводжується зміною самих факторів, сили і результативності їх впливу. А також призводить до варіації рівня досліджуваного явища в часі. В економіці дуже рідно зустрічаються чисто стаціонарні ряди, які не мають систематичних змін в середніх значеннях рівнів, їх дисперсіях, і ці характеристики не залежать від початку відліку часу. Тобто, як правило, економісти мають справу з тимчасовими рядами, що не є стаціонарними. Послідовність розташування досліджуваних даних у часі в таких рядах має істотне значення для

аналізу, адже, час - один з визначальних факторів для досліджуваного явища.

Прогнозування

Прогнозування часових рядів передбачає, що відомо значення деякої функції в перших n точках тимчасового ряду. Використовуючи цю інформацію необхідно спрогнозувати значення в n + 1 точці часового ряду. Існує безліч різних методів прогнозування, але на сьогоднішній день одними з найпоширеніших є метод Вінтерса і ARIMA модель.

Прогнозування є важливим елементом будь-якої інвестиційної діяльності. Для інвестора, що вкладає свій капітал з метою отримання прибутку, важливо знати, як будуть вести себе ті чи інші фінансові інструменти (об'єкти інвестування) в майбутньому, і від того, наскільки якісно буде виконано прогноз, безпосередньо залежатиме одержуваний їм прибуток. Ті, хто володіє більш досконалими методами аналізу і прогнозування фінансових часових рядів, будуть мати більш високу норму прибутку в порівнянні з тими, у кого її немає. Тому велике значення для інвестора має проблема вдосконалення цих методів.

Основна проблема в завданні аналізу і прогнозування полягає в побудові моделі, адекватно відображає динаміку фінансових часових рядів. Ринковий механізм, який характеризується великою кількістю постійно мінливих зв’язків, залежить від безлічі зовнішніх факторів, здатних суттєво вплинути на всю структуру його залежностей, причому вплив може бути найрізноманітнішим. Поява тих чи інших зовнішніх факторів не завжди відбивається в передісторії фінансового часового ряду, але може викликати значне порушення його динаміки. Саме в цьому полягає особливість практично всіх фінансових часових рядів. Ухвалення нового закону, зміна ставки рефінансування, відставка уряду, передвиборна кампанія, гучні скандали - ось лише невеликий список тих факторів, які здатні істотно вплинути на фінансовий ринок.

Для вирішення завдання аналізу і прогнозування фінансових часових рядів застосовується два основних підходи: теоретичний і практичний. Теоретичний підхід об'єднує моделі, гіпотези і теорії, що дають уявлення про найбільш загальні залежності ринкового механізму, представлених в деякому ідеалізованому вигляді. Серед них можна виділити наступні: теорія динамічного хаосу, модель ціноутворення на ринку капіталовкладень (CAPM), модель оцінки капітальних активів (APM). Дані роботи представляють скоріше теоретичний інтерес, ніж практичний, оскільки не дають конкретної моделі фінансового часового ряду, але дають рекомендації для її побудови.

Практичний підхід приділяє більше уваги безпосередньому моделюванню фінансових часових рядів, які розглядає як реалізацію деякої складної залежності невідомого виду. Головне в цьому підході те, що використовувана модель повинна успішно вирішувати завдання прогнозування, а вид її не має значення. Хоча, з іншого боку, для кращого розуміння залежностей бажано, щоб спосіб обчислення майбутніх значень фінансового часового ряду мав осмислене тлумачення. Вибір моделі, як правило, здійснюється емпіричним шляхом на основі деякого заданого універсального сімейства предикторів, можливості якого дозволяють описати будь-який часовий ряд. Вибір того чи іншого сімейства моделей відображає специфіку вирішення завдання прогнозування. Найбільш яскравим представником практичного підходу є технічний аналіз. Крім того, набули широкого поширення регресивні методи, а останнім часом - методи, засновані на wavelet-перетворення і нейронних мережах.

Загальна риса, яка об'єднує обидва підходи, полягає в спробі побудови єдиної моделі фінансового часового ряду. Однак зі сказаного вище випливає, що побудова такої моделі стикається з рядом принципових труднощів, обумовлених природою фінансового часового ряду.

4. Висновок.

Часовими ряди описують зміну тих чи інших характеристик у часі. Прикладів таких даних можна зустріти дуже багато - котирування валют, обсяги продажів, звернення клієнтів, дані в різних прикладних науках (соціологія, метеорологія, геологія, спостереження у фізиці) і багато іншого.

Часові ряди є поширеною і важливою формою опису даних, так як дозволяють спостерігати всю історію зміни того чи іншого значення. Це дає нам можливість судити про «типову» поведінку величини і про відхилення від такої поведінки.

Прогнозування розвитку фінансових показників є складним і водночас дуже важливим завданням. Від правильності прогнозу залежить величина прибутків або збитків інвестора. Показовим у цьому сенсі є фондовий ринок. При розробленні підходів до прогнозування фінансових часових рядів треба зважати на те, що переважна більшість учасників торгів на фондовому ринку є спекулятивними гравцями, які здійснюють свої вкладення з метою отримання максимального прибутку, а не мінімізувати середньоквадратичне відхилення, як це прийнято у випадку апроксимації функцій. І більшою мірою величина прибутку залежатиме саме від правильності передбаченого знаку зміни курсу, адже гравець фондового ринку отримує прибуток здебільшого від гри на пониження-підвищення. Тому є сенс налаштовувати прогнозуючу модель на передбачення напрямку зміни ціни, а не самого значення курсу.

Вивчення і розуміння методів аналізу часових рядів, уміння порівнювати їх і знаходити оптимальний для себе метод дає змогу правильно спрогнозувати свою діяльність в сфері економіки, торгівлі і виробництва; оптимізувати процеси, що протікають в суспільстві; приймати правильні рішення для бізнесу.

Список літератури

1. Чарівність хаосу/ Лоскутов А.Ю. - 2010. — 1329 с.
2. Застосування теорії часових рядів у економічних дослідженнях / Ковалева Г.Д. - 2008. — 56 с.
3. Эконометрика / Орлов А.І. - 2009. - 576 с.
4. Эконометрика. Учебник Для ВУЗов / Тихомиров Н.П. Дорохина Е.Ю. - 2009. - 512 с.
5. Аналіз часових рядів і прогнозування / Г.С. Кільдишев //А.А. Френкель - 1973. - 101 с.
6. Загальна теорія статистики/ Єлісеєва І.І.//Юзбашев М.М. - 2002.- 480 с.

УДК 621.391

ПОЛІЩУК А. О.,
ЗАДІРАКА В.К.

СТЕГАНОГРАФІЯ АУДІО ФАЙЛІВ ВИКОРИСТОВУЮЧИ МЕТОД НАЙМЕНШ ЗНАЧУЩОГО БІТУ З ПІДВИЩЕНОЮ ПОТУЖНІСТЮ

У цій статті основна увага спрямована на посилення забезпечення аудіо стеганографії, вводячи одну техніку кодування методом найменш значущого біту. Розроблюється метод високошвидкісного приховування даних аудіосистеми LSB із високою бітовою швидкістю, що зменшує спотворення вбудованого аудіо файлу з підвищеною ємністю секретного тексту. Використовуючи стандартний і запропонований алгоритм, біти прихованих даних вбудовуються в більш високий рівень LSB, що призводить до підвищеної стійкості до додавання шуму, що обмежується перцептивною прозорістю.

In this article, the focus is on strengthening the provision of audio steganography, introducing one technique of coding by the method of the LSB. The method of high-speed hiding of high-bit rate LSB audio data is developed, which reduces the distortion of the built-in audio file with the increased capacity of the secret text. Using the standard and proposed algorithm, bits of hidden data are embedded in a higher level of LSB, which results in increased resistance to the addition of noise, which is limited to perceptual transparency.

1. Вступ

Наукове дослідження в відкритій літературі почалося в 1983 р., коли Сіммонс заявив про проблему з точки зору спілкування в тюрмі. У своєму формулюванні двоє ув'язнених Аліса та Боб намагаються вилучити план втечі. Єдиний спосіб, яким вони можуть спілкуватися один з одним, - це через загальнодоступний канал, який ретельно контролюється наглядачем у в'язниці. Якщо палата виявляє будь-які зашифровані повідомлення чи код, він кине Алісу та Боба в одиночну камеру. Проблема стеганографії вводиться тоді; як Аліса і Боб готують план втечі, спілкуючись над громадським каналом таким чином, що Уорд не підозрює, що "щось незвичайне" відбувається. Зверніть увагу, як мета стеганографії відрізняється від класичної криптографії, яка полягає у приховуванні вмісту секретного повідомлення: стеганографія - це приховування самого існування таємного повідомлення.

Стеганографічні протоколи мають довгу та інтригуючу історію, яка походить від античності. Є історії секретних повідомлень, написаних невидимими чорнилами або прихованими в любовні листи (перший символ кожного речення може використовуватися, наприклад, для написання таємниці). Зовсім недавно, стеганографія використовувалася в'язнями та солдатами під час Другої світової війни, тому що всі листи в Європі були ретельно обстежені на той час. Поштові цензори викреслили все, що виглядало як конфіденційна інформація (наприклад, довгі рядки цифр), і вони переслідували осіб, чиї листи вважалися підозрілими. У багатьох випадках цензори навіть випадково видалили невинні висловлювання або цілі абзаци, щоб запобігти проникненню секретних повідомлень. Протягом останніх років стеганографія вивчалась в рамках інформатики, і було розроблено кілька алгоритмів для приховання секретних повідомлень у звичайних даних.

2. Метод найменш значущого біту

Метод найменш значущого біту (LSB) - найпростіший спосіб вставляти інформацію в цифровий аудіофайл. Підставляючи найменш значущий біт кожної точки відбору з двійковим повідомленням, метод LSB дозволяє кодувати велику кількість даних.

У методі LSB ідеальна швидкість передачі даних становить 1 кбіт / с на 1 кГц. Однак у деяких варіантах кодування LSB два найменш значущих біти зразка замінюються двома біт повідомлення. Це збільшує кількість даних, які можна кодувати, але також збільшує кількість шуму, що виникає, у аудіофайлі. Таким чином, слід враховувати вміст сигналу, перш ніж приймати рішення про використання методу LSB. Наприклад,

звуковий файл, який був записаний на шумній станції метро, маскує низькошвидкісний шум кодування. З іншого боку, той же звук буде звучати в звуковому файлі, що містить звуки фортепіано.

Щоб витягти секретне повідомлення з звукового файлу, закодованого методом LSB, одержувач потребує доступ до послідовності індексів вибірки, які використовуються в процесі вбудовування. Як правило, довжина секретного повідомлення, що кодується, менше, ніж загальна кількість зразків звукового файла. Потім слід вирішити, як вибрати підмножину зразків, що містять таємне повідомлення, та повідомити про це рішення одержувача. Одна методика полягає в тому, щоб почати з початку звукового файла та виконувати кодування LSB, поки повідомлення не буде повністю вбудоване, залишивши залишкові зразки без змін. Це створює проблему безпеки, однак в тому, що перша частина звукового файлу матиме різні статистичні властивості, ніж друга частина незмінного звукового файлу. Одне рішення цієї проблеми полягає в тому, щоб поставити секретне повідомлення з випадковими бітами, так що довжина повідомлення дорівнює загальній кількості зразків. Проте зараз процес введення закінчується зміною набагато більше зразків, ніж передача потрібного секрету. Це підвищує вірогідність того, що злочинці будуть підозрювати секретне спілкування. Більш витончений підхід полягає у використанні генератора псевдовипадкових чисел, щоб розповсюджувати повідомлення над звуковим файлом у випадковому порядку.

Одним з популярних підходів є використання випадкового інтервального методу, в якому секретний ключ, яким має відправник, використовується як насіння у генераторі псевдовипадкових чисел для створення випадкової послідовності індексів вибірки. Приймач також має доступ до секретного ключа та знань генератора псевдовипадкових чисел, що дозволяє відновлювати випадкову послідовність показників вибірки. Однак, для того, щоб генератор псевдовипадкових чисел не генерував один і той же індекс вибірки два рази, слід встановити перевірки. Якщо це сталося, виникне зіткнення, коли зразок, уже змінений частиною повідомлення, буде змінено знову. Проблему зіткнень можна подолати, відстежуючи всі вже використані зразки. Інший підхід полягає у розрахунку підмножини зразків за допомогою псевдовипадкової перестановки всього набору за допомогою безпечної хеш-функції. Ця методика гарантує, що один і той же індекс ніколи не генерується більше одного разу.

2.1 Звичайний метод найменш значущого біту

Хування даних у найменш значущих бітах (LSB) у часовій області є одним з найпростіших алгоритмів з дуже високою швидкістю передачі додаткової інформації [2]. Приховані дані зазвичай вибирає підмножина всіх доступних звукових зразків хоста, обраних секретним ключем. Операція заміни на LSB виконується на цьому підмножині, де біти, які слід приховати, замінять початкові значення біт. Процес вилучення просто витягує приховані дані, читаючи значення цих бітів із звукового об'єкту. Отже, декодер потребує всіх зразків аудіо, які використовувалися під час вбудовування. Випадковий вибір зразків, використовуваних для вбудовування, означає низькоенергетичний адитивний білий гауссовий шум (AWGN). З літератури добре відомо, що людська слухова система (HAS) дуже чутлива до AWGN. Цей факт обмежує кількість бітів, які можуть бути непомітно змінені під час вбудовування прихованих даних.

Основною перевагою методу LSB є дуже висока швидкість каналу; використання лише одного біту аудіовипромінювання хоста має потужність 44,1 кбіт/с (частота дискретизації 44 кГц, всі зразки, використовувані для приховування даних) і низька обчислювальна складність. Очевидним недоліком є значно низька надійність, через те, що прості випадкові зміни бітів знищують закодовані приховані дані.

Оскільки кількість використовуваних бітів під час кодування LSB збільшується або, рівнозначно, збільшується глибина модифікованого LSB-шару, збільшується ймовірність статистичного виявлення вбудованого повідомлення та знижується прозора сприйняття стемо-об'єктів. Тому існує межа для глибини використовуваного шару LSB у кожному зразку аудіо-хосту, який може використовуватися для приховування даних. Суб'єктивний тест прослуховування показав,

що в середньому максимальна глибина LSB, яку можна використовувати для приховання даних на основі методу LSB, не викликаючи помітних перцептивних спотворень, є четвертим шаром LSB, коли використовуються 16 біт на одиницю аудіоверсії зразка. Тести виконувалися з великою колекцією звукових зразків та окремими особами з різним фонові та музичним досвідом. Жоден з перевірених аудіохірургічних послідовностей не мав перцептивних артефактів, коли четвертий LSB був використаний для приховування даних, хоча в певних стилях музики ця межа навіть вище, ніж четвертий шар LSB. Надійність прихованих даних, вбудованого за допомогою методу кодування LSB, зростає з збільшенням глибини LSB, що використовується для приховування даних. Тому покращення стійкості даних, отриманого шляхом збільшення глибини використовуваного шару LSB, обмежується сприйнятливою прозорістю, що є четвертим шаром LSB для стандартного алгоритму кодування LSB.

2.2 Модифікований метод найменш значущого біту

Цей спосіб дозволяє зміщувати межу для прозорих даних, які ховаються в аудіо з четвертого шару LSB до шести рівня LSB, використовуючи двостадійний підхід [4]. На першому кроці біт приованих даних вставляється в і-й шар LSB аудіо-хоста за допомогою методу кодування LSB. На другому кроці імпульсний шум, спричинений вкладанням даних, формується для зміни властивостей білого шуму. Стандартний спосіб кодування LSB просто замінює початковий звуковий біт в і-му шаблоні (i = 1, ..., 16) з біт з потоку бітових даних. Ключовою ідеєю запропонованого алгоритму LSB є вбудовування бітної води, що викликає мінімальне спотворення вбудованого аудіо-хоста. Зрозуміло, що, якщо тільки один з 16 біт у зразку фіксований, інші біти можуть бути перевернуті, щоб мінімізувати помилку вкладання. Проте алгоритм вилучення залишається незмінним; він просто витягує приховані дані, читаючи бітові значення з попередньо визначеного шару LSB. У алгоритмі вкладання (I + 1) LSB шар спочатку змінюється шляхом вставки поточного біту повідомлення. Потім задається алгоритм, наведений нижче. У випадку, якщо біт не потрібно взагалі модифікувати через те, що він вже має правильне значення, з цим зразком сигналу не вживаються жодні дії.

Щоб сховати повідомлення у зразку хвилі, зачепивши одну одиницю носія, поміститься один біт повідомлення в найнижчий 4-й біт носія, решта перевернеться і замінить блок на потік призначення. Кодування LSB пояснюється наступною процедурою:

- читання одного зразоку з хвильового потоку.
- отримання наступного біту з поточного байта повідомлення.
- переміщення його в поточний 4-й біт зразка.
- перевернення інших 3 бітів відповідно.
- копіювання решти хвилі без змін.

3. Метод найменш значущого біту з підвищеною потужністю

У модифікованій техніці кодування LSB ми бачили, що 4-й біт встановлюється відповідно до секретного повідомлення. Якщо біт вибірки не дорівнює бітові секретних повідомлень, ми просто перевертаємо решту бітів цього заданого зразка. У нашій запропонованій моделі ми беремо послідовні два біти з секретного повідомлення, і замість того, щоб змінити один біт у зразку, ми змінюємо два біти (4-а та 3-а позиції) зразка. Якщо в цих двох біт змінюється, ми перекриваємо решту LSB, інакше не буде змін.

Отже, можемо сказати, що з тією ж бітовою помилкою ми збільшили можливості вибірки, щоб приховати секретне повідомлення більшого розміру.

Запропонована методика LSB пояснюється наступною процедурою:

- читання один зразок з хвильового потоку.
- отримання наступних двох бітів від поточного байта повідомлення.
- переміщення його в поточний 4-й та 3-й біти зразка.
- обернення решти 2 біта відповідно.
- копіювання решти хвилі без змін.

Зрозуміло, що пропонований спосіб вводить меншу помилку і вищу ємність під

час введення прихованих даних. Секретне повідомлення повністю вбудоване, коли ми намагаємося вбудувати два бітових значення, але коли ми вбудовуємо 1 біт значення, останні 4 біти не отримують жодного зразка для вставки. Якщо використовується 4-й шар LSB, абсолютне значення помилки коливається від 1 до 4 QS, тоді як стандартний метод у тих самих умовах призводить до постійної абсолютної похибки 8 QS. Отже, середня потужність введеного шуму становить на 9,31 дБ менше, якщо використовуватиметься запропонований метод кодування LSB. На додаток до зменшення об'єктивної якості вимірювання, вираженого як значення співвідношення сигнал / шум (SNR), запропонований спосіб вводить на другому етапі вкладання формування шумів для збільшення прозорості сприйняття методу. Подібна концепція, яка називається методом дифузії помилок, зазвичай використовується для перетворення справжніх кольорових зображень на кольорові зображення на палітрі. У нашому алгоритмі помилка вставки поширюється на чотири послідовні зразки, оскільки зразки, що є попередниками поточного зразка, не можуть бути змінені, оскільки інформаційні біти вже вбудовані в їхні біти.

Нехай e (n) позначає помилку вкладання зразка a (n). Для випадку вставки в 4-й шар LSB наступні чотири послідовні зразки аудіо-хосту модифікуються відповідно до цих виразів:

$$a(n+1) = a(n+1) + be(n)c$$
$$a(n+3) = a(n+3) + be(n)c/3$$
$$a(n+2) = a(n+2) + be(n)c/2$$
$$a(n+4) = a(n+4) + be(n)c/4,$$

де bAc означає операцію округлення A до найближчого цілого числа, меншого чи рівного A. Помилка дифузії формує вхідний імпульсний шум, введений вкладанням LSB, шляхом розмивання та зміни його розподілу на перцептуально краще. Ефект найбільш підкреслюється під час тихого періоду аудіосигналу та у фрагментах з низькою динамікою, наприклад широкі мінімуми або максимуми. Обидва кроки вкладання сукупно підвищують суб'єктивну якість звукового об'єкта. Тому ми очікуємо, що, використовуючи запропонований двоступінковий алгоритм, ми можемо збільшити глибину розміщення секретних даних далі, ніж 4-й шар LSB, і відповідно збільшити надійність алгоритму відносно появи шуму.

4. Результати досліджень

Модифікований алгоритм методу найменш значущого біту був протестований на 7 аудіофайлах з різних стилів музики (поп, рок, техно, джаз). Звукові витримки були відібрані так, щоб вони представляли широкий спектр музичних жанрів, тобто аудіозаписи з різними динамічними та спектральними характеристиками. Всі музичні фрагменти отримали приховані дані за допомогою запропонованого та модифікованого алгоритму. Частота 44,1 кГц, вибіркові аудіофайли, представлені 16 бітами на вибірку. Тривалість зразків коливалася від 10 до 15 секунд. Як визначено вище, співвідношення сигнал/шум для вбудованх даних обчислюється так:

$$SNR = 10 \not\in \log 10\Sigma\, n\, x2\,(n) / \Sigma\, n\, [x2\,(n) - y2\,(n)]$$

де x (n) представляє вибірку входу аудіопослідовність і y (n) означає вибірку звуку з модифікованими LSB.

Аудіо файл	Швидкість передачі	Кількість помилок (1 біт)	Кількість помилок (2 біт)	Частота бітової помилки (1 біт)	Частота бітової помилки (2 біт)
1	64	26	19	0,0813	0,0594
2	64	20	30	0,0625	0,0938
3	64	40	08	0,1250	0,0250
4	88	45	11	0,1406	0,0344
5	88	24	24	0,0750	0,0750
6	176	37	14	0,1156	0,0437
7	176	26	17	0,0813	0,0531
8	176	40	12	0,1250	0,0375

Результати суб'єктивних тестів показали, що частота бітових помилок висока, коли ми використовуємо зразок звуку 64 Кбіт/сек для запропонованого LSB. Але дивлячись на файли з вищим значенням швидкості передачі у Кбіт/сек, то швидкість передачі даних менша, і вона помітно зведена до мінімуму.

5. Висновки

Був проведений експеремент вкраплення прихованих даних, використовуючи модифікований метод найменш значущого біту для аудіоданих. Ідея алгоритму полягає в тому, що вбудовуються два біти прихованих даних, що забезпечує мінімальне спотворення аудіо-файлу з фіксованою довжиною з високою ємністю. Прослуховуючий тест показав, що описаний алгоритм успішно приймає дві бітові позиції для вкладання секретних даних, не впливаючи на прозорість сприйняття звукового сигналу. Поліпшення потужності при наявності аддитивного шуму є очевидним, оскільки запропонований алгоритм отримує значно нижчі бітові помилки, ніж стандартний алгоритм з інтелектуальною кадровою роботою, де шар буде автоматично обраний системою, і одна позиція використовується для приховування дані, тому ми не потребували більшої кількості вибірки, щоб приховати секретний текст.

Стеганографія має ряд недоліків у порівнянні з шифруванням. Це вимагало багато накладних витрат, щоб приховати відносно мало бітів інформації. Як тільки система буде виявлена, вона стає фактично марною. Таку проблему можна подолати, якщо метод вставки залежить від певного ключа. Крім того, повідомлення може бути спочатку зашифровано, а потім приховано за допомогою стеганографії.

Список літератури

1. Проблемы выявления скрытой передачи информации по сетям [Електронний ресурс] / Режим доступу: http://www.ipages.ru/index.php?ref_item_id=798&ref_dl=1.
2. Стеганография в XXI веке. Цели. Практическое применение. Актуальность [Електронний ресурс] / Режим доступу: https://habrahabr.ru/post/253045/
3. Грибунин В.Г., Оков И.Н., Туринцев И.В Цифровая сеганография [Текст] / Грибунин В.Г. // Москва: СОЛОН-Пресс, 2002 р. – 261 с.
4. Phitzman B. Information hiding terminology [Текст] Information Hiding: First Int. Workshop “InfoHiding’96” / Springer as Lecture Notes in Computing Science, 1996 р. – С. 350-374.

УДК 004.93(015.7)

ПРОХОРОВА К.С.,
ГУЛЯНИЦЬКИЙ Л.Ф.

РОЗВ`ЯЗУВАННЯ ЗАДАЧІ КОМАНДНОГО СПОРТИВНОГО ОРІЄНТУВАННЯ З ЧАСОВИМИ ВІКНАМИ

Наведено математичну постановку задачі командного спортивного орієнтування з часовими вікнами. Застосовано метаевристичні методи для розв`язання даної задачі: повторюваний локальний пошук, алгоритм імітації відпалу. Проведено обчислювальний експеримент для оцінки роботи застосованих алгоритмів.

Ключові слова: ЗАДАЧА СПОРТИВНОГО ОТІЄНТУВАННЯ, ЗАДАЧА КОМАНДНОГО СПОРТИВНОГО ОРІЄНТУВАННЯ З ЧАСОВИМИ ВІКНАМИ, МЕТАЕВРИСТИКА, ЛОКАЛЬНИЙ ПОШУК, ПОВТОРЮВАНИЙ ЛОКАЛЬНИЙ ПОШУК, АЛГОРИТМ ІМІТАЦІЙНОГО ВІДПАЛУ, ЗАДАЧА МАРШРУТИЗАЦІЇ ТРАНСПОРТНИХ ЗАСОБІВ

The mathematical model for team orienteering problem with time windows is given. Metaheuristics are used to solve this problem: repetitive local search, simulation annealing. The computational experiment was conducted to evaluate the work of the used algorithms.

Keywords: ORIENTEERING PROBLEM, TEAM ORIENTEERING PROBLEM WITH TIME WINDOWS, METAHEURISTIC, LOCAL SEARCH, ITERATED LOCAL SEARCH, SIMULATED ANNEALING ALGORITHM, VEHICLE ROUTING PROBLEM

1.Вступ

Наразі існує велика кількість задач, що належать до класу Vehicle routing problem (VRP), багато проблем з різних предметних областей можуть бути змодельовані у термінах задач цього класу. Через це, їх добре досліджено. Проте, існують проблеми, що можуть бути сформульовані як в термінах задач VRP, так і в термінах Orienteering problem (OP), причому формулювання у термінах OP дозволяє краще врахувати специфіку предметних областей даних задач. Тому, наразі активно проводяться дослідження задач класу OP, виникають нові методи розв`язування [1] і нові постановки задач [2].

Потреба у дослідження задач цього класу зростає з появою нових практичних застосувань. Наприклад, популярності набуває створення програмного забезпечення для складання туристичних маршрутів [3], розподілу медичного персоналу [4] і для краудсорсингу [5]. Проблеми такого типу добре моделюються у термінах однієї з задач класу OP – Team orienteering problem with time windows (TOPTW). Слід зазначити, що можливо обрати інші математичні формулювання задачі: VRP чи складніші модифікації TOPTW. Вибір зроблено на користь TOPTW, бо математична постановка у термінах VRP не враховує специфіку проблем (наприклад, той факт, що не всі візити обов`язкові), а подальші модифікації TOPTW приводять до ускладнень математичної постановки і методів розв’язку даної проблеми.

2.Математична постановка задачі

TOPTW – задача максимізації сумарної корисності набору маршрутів, що обмежені у часі.

Пункти, що треба відвідати, можна представити як вершини орієнтовного повного зваженого графу. Граф містить $n, n \in N$ вершин. Необхідно побудувати $m, m \in N$ маршрутів у цьому графі. Маршрути обмежені у часі. T_{max} – максимальна тривалість кожного з маршрутів.

Кожен маршрут починається з вершини $i = 1$, а закінчується в вершині $i = n$. При відвідуванні вершини корисність маршруту збільшується на величину $Score_i$. Дане збільшення загальної корисності маршруту відбувається лише при першому відвідуванні вершини $i \in \overline{1, n}$. Відвідати певну вершину можливо лише у рамках її часового проміжку(вікна). Дане вікно обмежене величинами: O_i – початок часового вікна, коли можна відвідати пункт, C_i – кінець часового вікна, коли можна відвідувати пункт $i \in \overline{1, n}$. Необхідно зазначити, що відвідування

певної вершини можна почати в час C_i. T_i – час необхідний на відвідування пункту $i \in \overline{1,n}$. c_{ij} – час, що необхідний на переміщення між пунктами $i \in \overline{1,n}$ та $j \in \overline{1,n}$. Для формулювання математичної постановки вводяться додаткові змінні і константи. s_{id} – час початку візиту до пункт $i \in \overline{1,n}$ на маршруті $d \in \overline{1,m}$. x_{ijd} – змінна, що приймає значення 1, тільки коли на маршруті $d \in \overline{1,m}$ пункт $j \in \overline{1,n}$ відвідується одразу за пунктом $i \in \overline{1,n}$. M – константа, значення якої велике відносно задіяних даних. Вихідна змінна y_{id}, що позначає відвідування певної вершини на маршруті визначається так:

$$y_{id} = \begin{cases} 1, \text{якщо на маршруті } d \in \overline{1,m} \\ \text{відвідується пункт } i \in \overline{1,n} \\ 0, \text{ в іншому випадку} \end{cases}$$

Цільова функція, що позначає сумарну корисність від відвідування всіх пунктів, що включаються в маршрути, має такий вигляд:

$$\sum_{d=1}^{m}\sum_{i=2}^{n-1} Score_i y_{id} \to \max$$

$$y_{id} \in \{0,1\}\ i \in \overline{1,n}, d \in \overline{1,m}$$

Початковий і кінцевий пункти повинні бути відвідані рівно m разів:

$$\sum_{j=2}^{n-1}\sum_{d=1}^{m} x_{1jd} = \sum_{i=2}^{n-1}\sum_{d=1}^{m} x_{ind} = m$$

$$x_{ijd} \in \{0,1\}\ i,j \in \overline{1,n},\ d \in \overline{1,m}$$

Потрібно врахувати правило збереження потоку, а саме те, що пункт відвідується і покидається однакову кількість разів.

$$\sum_{i=1}^{n-1} x_{ikd} = \sum_{j=2}^{n} x_{kjd} = y_{kd}$$

$$k \in \overline{2,n-1}, d \in \overline{1,m}$$

Нижченаведена формула забезпечує, що проміжок часу між двома послідовними пунктами достатньо довгий, щоб виділити час на відвідування першого з них і переміщення між першим і другим пунктами.

$$s_{id} + T_i + c_{ij} - s_{jd} \le M(1 - x_{ijd})$$

$$i,j \in \overline{1,n}, d \in \overline{1,m}$$

Таким чином формулюється обмеження, що пункт не буді відвідано більше разу на всіх маршрутах:

$$\sum_{d=1}^{m} y_{kd} \le 1\ k \in \overline{2,n-1}$$

Загальна тривалість кожного маршруту не більше T_{max}, тому математична постановка містить обмеження такого вигляду:

$$\sum_{i=1}^{n-1}(T_i y_{id} + \sum_{j=2}^{n} c_{ij} x_{ijd}) \le T_{max},\ d \in \overline{1,m}$$

Початок відвідування певної вершини повинен відбуватись у рамках пвного часового вікна, тому вводяться такі обмеження:

$$O_i \le s_{id}\ i \in \overline{1,n}, d \in \overline{1,m}$$

$$s_{id} \le C_i\ i \in \overline{1,n}, d \in \overline{1,m}$$

$C_n = T_{max}$, бо кінцевий пункт можна відвідати не пізніше T_{max}. $T_1 = T_n = 0$, бо кінцевий і початковий пункти не потребують часу для відвідування.
розв'язком задачі є:

- розбиття множини вернин графу на підмножини (маршрути);
- задання порядку обходу на кожній підмножині.

Розв'язок є прийнятним (допустимим), якщо всі маршрути задовольняють обмеженням задачі.

3.Метод розв`язання задачі

За своєю складністю задачі TOPTW не менш складна ніж TOP. В свою чергу, TOP є модифікацією задачі OP (OP є частковим випадком TOP) тому, як і OP, TOP і TOPTW є NP-складними задачами. Доведення того, що OP є NP-складною задачею наведено у [6].

Задача TOPTW може бути оптимально розв'язана точними методами. Втім, такі методи можуть використовуватись лише для розв'язку задач із невеликою кількістю вершин. До цих методів відносяться динамічне програмування і метод гілок та меж. Наприклад, у роботі [7] TOPTW розв`язано точними методами для графу з 30 вершинами.

Зважаючи на складність проблеми, переважна більшість літератури присвячена розв`язанню проблеми неточними методами.

Для розв`язку даної задачі застосовано два ефективні і добре відомі алгоритми: алгоритм повторюваного локального пошуку, алгоритм імітації відпалу. В їх

основі лежить однакова процедура локального пошуку [8].

Побудова розв'язку починається з того, що кожен з $d \in \overline{1,m}$ маршрутів містить початкову і кінцеву вершини. Для кожної вершини, що не належить до жодного маршруту, визначається величина $shift_i$ $i \in \overline{1,n}$. Дана величина позначає те, наскільки зміститься у часі прибуття до вершини перед якою буде розміщена нова вершина. Потім, коли для кожної вершини визначена найкраща позиція, робиться оцінка $Score_i^2/shift_i$. Дана оцінка одночасно має на меті побудову коротких маршрутів між вершинами і максимізацію сумарної корисності всіх маршрутів. Обирається вершина з найбільшою оцінкою і пошук повторюється знов. Робота локального пошуку завершується, коли через обмеження в часі маршрутів неможливо додати нову вершину. Після цього проводиться збурення, що являє собою вилучення вершин з маршруту. Потім знову виконується процедура локального пошуку. Отримане значення цільової функції може бути меншим за попереднє, тоді старий розв'язок замінює поточний. У протилежному випадку поточний розв'язок стає новим і знову відбувається збурення. Алгоритм завершує свою роботу, після наперед визначеної кількості ітерацій.

Для того, щоб скоротити час роботи, обчислення околу при роботі процедури локального пошуку відбувається паралельно. А саме, для кожної вершини, що не входить в маршрути, незалежно обчислюється найкраща позиція для розміщення.

Алгоритм імітації відпалу, як і повторюваний локальний пошук, використовує модифікований локальний пошук, що описано вище. Для алгоритму імітації відпалу початкову температуру визначено за принципом, що розглянуто у роботі [9]. Імовірність прийняття розв`язку який погіршує значення цільової функції розраховується згідно розподілу Больцмана, а розклад охолодження визначається на основі геометричного закону з константою 0.95.

4.Обчислювальний експеримент

Вхідні дані, що однакові для всіх алгоритмів:

- будується 2 маршрути;
- виконується 20 ітерацій алгоритму.

Використовувалось таке апаратне забезпечення: процесор Intel Core i5-3317U, 1.7 GHz з оперативною пам`яттю 4 GB. В таблиці 1 наведено результати роботи алгоритму повторюваного локального пошуку з застосуванням модифікованого і не модифікованого локального пошуку.

Було проведено 5 прогонів кожного з алгоритмів і розраховано середнє значення часу їх роботи.

Для роботи процедури імітації відпалу початкова температура буле розрахована згідно процедури, що наведена у роботі [9]. На вхід процедури подається очікувана початкова імовірність прийняття розв`язків, що мають гірше значення цільової функції. В даному випадку обирається значення 0.8.

Табл.1 – усереднений час роботи алгоритму повторюваного локального пошуку

Кількість вершин	48	72	96	144	160	192	216	240	288
Час роботи ЛП, с	4.97	21.45	27.8	47.72	57.77	78.72	89.272	156.9	167.48
Час роботи модифікованого ЛП, с	3.23	12.3	21.03	39.56	44.43	69.33	70.8	108.8	127.7
Покращення, %	35	42.6	24.34	17.09	23.09	11.92	20.69	30.61	23.75

На рисунку 1 наведено результати роботи алгоритмів повторюваного локального пошуку і алгоритму імітації відпалу. Графік показує залежність розмірності задачі від значення цільової функції.

5.Інтерпритація результатів

З огляду на те, що процедура обчислення околу в локальному пошуку є досить трудомісткою, то застосування паралельних обчислень дає покращення у швидкості роботи алгоритмів, тому таку модифікацію можна вважати доречною.

Розглянуті алгоритми дають досить подібні результати, але згідно

проведеному обчислювальному експерименту, алгоритм повторюваного локального пошуку знаходить кращі розв`язки. Дана ситуація може змінитись, якщо збільшити кількість ітерацій. Проте, треба зазначити, що задача TOPTW розв`язується щоб стати частиною веб-застосувань, чи мобільних застосувань. Тому треба віддавати перевагу алгоритмам, які дають хороші розв'язки якнайшвидше, а не забезпечують диверсифікацію у просторі пошуку розв`язків. Крім того, оцінка початкової температури досить трудомістка процедура, яку треба проводити перед початком роботи алгоритму на новому наборі даних. Тому у більшості випадків можна віддати перевагу процедурі повторюваного локального пошуку.

6.Висновок

Наведено і описано математичну постановку задачі командного спортивного орієнтування з урахуванням часових вікон. Задачу розв`язано з використанням алгоритмів повторюваного локального пошуку та імітації відпалу. Проведено обчислювальний експеримент для оцінки роботи застосованих алгоритмів. Порівняно результати роботи алгоритмів.

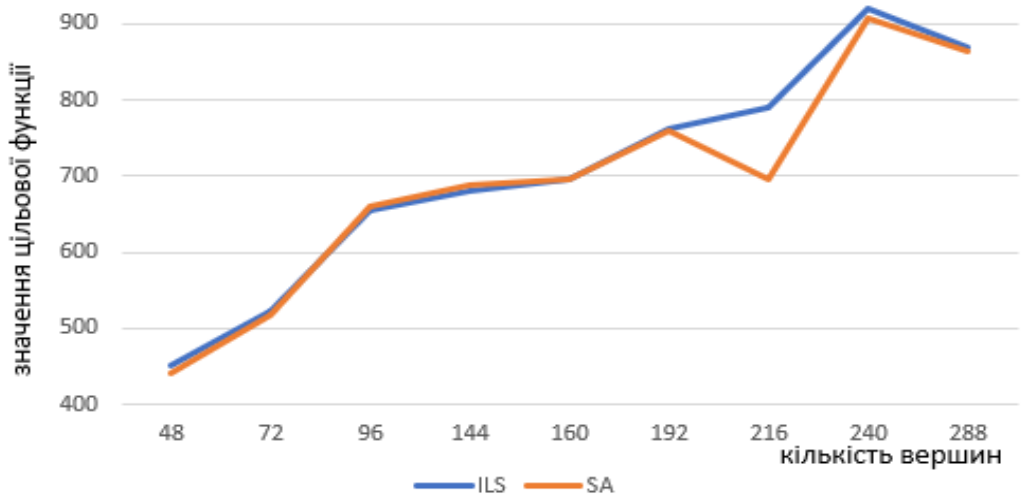


Рис.1 – результати роботи алгоритмів повторюваного локального пошуку і імітації відпалу

Список літератури

1. Vansteenwegen P. The orienteering problem: A survey / P. Vansteenwegen, D. Oudheusdena, W. Souffriau. // 209. – 2011. – №1. – С. 1–10.

2. Kara I., Derya T., Bicakci P. New Formulations for the Orienteering Problem // Procedia Economics and Finance. – 2016. – №39. – С. 849–854.

3. Gavalas D., Mastakas K., Charalampos K., Pantziou G. A survey on algorithmic approaches for solving tourist trip design problems // Journal of Heuristics. – 2014. – №20. – С. 291–328.

4. Tackling large-scale home health care delivery problem with uncertainty. / C.Chen, Z. Rubinstein, S. Smith, H. Lau. // Twenty-Seventh International Conference on Automated Planning and Scheduling. – 2017. – №27. – С. 18–23.

5. TRACCS: A Framework for Trajectory-Aware Coordinated Urban Crowd-Sourcing / [C. Chen, S. Cheng, A. Gunawan та ін.]. // HCOMP. – 2014.

6. Golden L. The orienteering problem / L. Golden, R. Vohra, L. Levy. // Naval Research Logistics. – 1987. – №34. – С. 307–318.

7. Kim B. A Branch-and-Price Approach for the Team Orienteering Problem with Time Windows / B. Kim, H. Tae. // The International Journal of Industrial Engineering: Theory, Applications and Practice. – 2015. – №22. – С. 243–251.

8. Iterated local search heuristic for the team orienteering problem with time windows / P.Vansteenwegen, W. Souffriau, G. Berghe, D. Oudheusden. // Computers & Operations Research. – 2009. – №36. – С. 3281–3290.

9. Walid B. Computing the Initial Temperature of Simulated Annealing / Ben-Ameur Walid. // Computational Optimization and Applications. – 2004. – №29. – С. 369–385.

УДК 621.391

РОМАНЧУК Р. О.,
ЗАДІРАКА В.К.

ПРАКТИЧНЕ ПОКРАЩЕННЯ СТЕГАНОСТІЙКОСТІ МЕТОДУ НАЙМЕНШ ЗНАЧУЩОГО БІТУ ШЛЯХОМ МОДИФІКАЦІЇ СХЕМОЮ РОЗПОДІЛУ СЕКРЕТУ ШАМІРА ДЛЯ ЦИФРОВИХ КОНТЕЙНЕРІВ

У цій статті стеганографічний метод та схема розподілу секрету зображень були використані разом для надійного збереження секретного ключа, який використовується в алгоритмі шифрування AES. По-перше, секретний ключ вбудований в зображення за допомогою технології LSB, далі стеганоконтейнер розділяється на зображення за допомогою схеми розподілу секрету Шаміра. Досить важко отримати дещо значуще з будь-якою з цих часток зображення. Але приховані дані можуть бути відновлені порогово-розподіленими зображеннями. В результаті, ефективність роботи обох методів була продемонстрована з точки зору безпеки ключа шифрування.

In this publication, one of the steganographic methods with secret image sharing used in a pair to safely secret key storage, which used in the AES cryptoalgorithm. First of all, the secret key inserted in image using LSB technology, after that the steganocontainer divided into images using the Shamir's algorithm of image secret sharing. It is difficult to get somewhat significant with any of these particles in the picture. But covert data can be restored by threshold-distributable images. In result, the performance of these methods showed in terms for encryption key security.

1. Вступ

Сьогодні, при повсякденному використанні Інтернету, важливість інформаційної безпеки ще більше підвищилася. Для забезпечення інформаційної безпеки використовуються різні методи і протоколи. Найбільш поширеними методами захисту є криптографія і стеганографія. У криптології вміст даних змінюється за допомогою ключа, в стеганографії вміст не змінюється, але дані сховані в стеганоконтейнері. Таким чином, мета полягає в тому, щоб приховане повідомлення не було повідомлено третіми особами. Згідно з методами, що використовуються в криптографії, дуже важливо, щоб ключ шифрування зберігався в секреті. Цей ключ зазвичай зберігається або на комп'ютері, або на будь-якому магнітному носії. При розробці методів шифрування виникає проблема того, що ключ захищений або переданий надійно. У літературі були запропоновані схеми секретного обміну як рішення цих проблем. Вперше концепція схеми секретного обміну була запропонована Шаміром і Блеклі в 1979 році. У той же час схеми секретного обміну відомі як порогова схема (k, n). Порогова схема A (k, n) дозволяє секрету ділитися між n учасниками таким чином, що будь-які k з них може відновити секрет, але будь-які k-1 або менше не мають абсолютно ніякої інформації про таємницю.

Метод секретного обміну Шаміра – це порогова схема, заснована на поліноміальній інтерполяції Лагранжа. Блеклі використовував секретний поділ як кінцеву геометрію, засновану на гіперплощинах в k-вимірному просторі. Пізніше, залишилися теоретичні схеми розподілу порогових секретних даних, що використовують властивості теорії чисел, були знайдені в 1983 році Міньоттом і Асмутом з Блумом. Ці схеми використовують спеціально відібрані цілі масиви згідно китайської теореми про залишки.

Виникала проблема забезпечення надійної передачі даних, які повинні зберігатися в секреті у військовій, комерційній та медичній областях, таких як електронна інформація, зображення, тексти і звуки, які використовуються спільно з використанням Інтернету. Проведена велика робота по забезпеченню безпеки секретних зображень.

Візуально схеми секретного розподілу були спочатку запропоновані Наором і Шаміром в 1994 році. У цій схемі (k, n), як і в пороговій схемі, приховані образи

застосовуються візуально до криптографічних технік для генерації n нічого не значущих частин і розподіляють серед учасників, які поділяють цей секрет. Щоб відновити прихований образ, має бути накладене не менш ніж k зображень учасника. Якщо кількість учасників менше k, інформація про приховане зображення не може бути отримана. Коли вихідне зображення порівнюється з прихованим зображенням, на етапі відновлення губиться контраст, оскільки розмір розподілів вдвічі менше прихованого розміру зображення [1,2].

2. Метод найменш значущого біту

Стеганографія – це мистецтво та наука про невидиме спілкування. Це досягається шляхом приховування інформації в іншій інформації, що приховує факт передачі інформації. У стеганографії інформація приховується в зображеннях, аудіо, відео тощо. Система стеганографії складається з трьох елементів: контейнер (приховує таємне повідомлення), секретне повідомлення та стеганоконтейнер (який є контейнером з вкрапленим повідомлення).

Секретне повідомлення сховане в бітах кольорів пікселя зображення контейнера. Кольоровий малюнок складається з пікселів, які відповідають рівню червоного, зеленого та синього. Червоні, зелені та сині кольори можуть бути об'єднані, щоб створити всі кольори. Кожен колір знаходиться в діапазоні 0-255, і кожен колірний піксель складається з 24 [3].

Вставка найменшого значущого біта – це звичайний, простий підхід до вбудовування інформації в контейнер. Останній біт кожного пікселя підмінюється бітом секретного повідомлення. Під час використання 24-бітового зображення можна використати кожен з компонентів червоного, зеленого та синього кольорів, оскільки кожен із них представлений байтом. Іншими словами, кожний піксель може зберігати 3 біти. Таким чином, зображення 256 × 256 пікселів може зберігати загальну кількість 196 608 бітів вкраплених даних.

Наприклад, сітка 3 пікселів для 24-бітового кольорового зображення може бути такою:

контейнер:

10101111 00011000 11000010 (175,24,194)
10110000 00010110 11001010 (176,22,200)
10100100 00011000 11000100 (180,24,196)

Секретними даними буде ‘a’: ASCII значення ‘a’ 97 = 01100001

Після застосування методу найменш значущого біту стеганоконтейнер виглядатиме

10101110 00011001 11000011 (174,25,195)
10110000 00010110 11001010 (176,22,200)
10100100 00011001 11000100 (180,25,196)

3. Схема розподілу Шаміра

Ідея, на якій заснована схема Шаміра, полягає в тому, що для інтерполяції многочлена ступеня k-1 потрібно k точок. Якщо відомо меншу кількість точок, то інтерполяція буде неможливою. Позначимо:

p - велике просте число (більше будь-якого секрету M, який передбачається розділяти в цій схемі). Тоді $M \in Z_p$: n - число часток секрету; k - мінімальний розмір дозволеної групи.

Роботу алгоритму можна розділити на 3 етапи.

Підготовчий етап.

Дилер вибирає випадковим чином коефіцієнти $S_1, S_2, S_3, ..., S_{k-1} \in Z_p$ і складає секретний многочлен

$$S(x) = S_{k-1}X^{k-1} + S_{k-2}X^{k-2} + ... + S_1X + M \bmod p$$

де M – розділяючий секрет, а інші коефіцієнти - довільні елементи поля (коефіцієнти многочлена дилер зберігає в таємниці). очевидно, що S(0) = M. Далі дилер обирає n різних несекретних ненульових елементів $r_1, r_2, r_3, ..., r_n$ із Z_p, кожен з яких ставить у відповідність одному учаснику схеми.

Розподіл секрету.

Дилер обчислює значення многочлена $c_1 = S(r_1), c_2 = S(r_2), ..., c_n = S(r_n)$. Частка кожного користувача A_i - це пара чисел $(r_i, c_i), i = 1,2,...,n$. Частки роздають учасникам схеми.

Відновлення секрету.

Щоб відновити секрет, треба скористатися інтерполяційною формулою Лагранжа: якщо потрібно побудувати многочлен S(x) ступеня k-1, який при $x_1, x_2, ..., x_k$ приймає відповідно значення $y_1, y_2, ..., y_k$, то цим многочленом буде: $S(x) = \sum_{j=0}^{k-1} y_i \prod_{j \neq i} \frac{x - x_j}{x_i - x_j}$. k як в схемі розподілу секрету многочлен належить обрати так, щоб S (0) = M, то з формули Лагранжа слідує:

$$M = \sum_{i=0}^{k-1} c_i S_i, \partial e : S = \prod_{j \neq i} \frac{r_j}{r_j - r_i}$$

З описаного вище, стає ясно, що для більших значень порога, обчислення стає повільнішим.

4. Схема типової стеганосистеми

Основними поняттями у стеганографії є повідомлення та контейнер. Повідомлення $m \in M$ – певна закрита інформація, яку необхідно приховати. $M = \{m_1, m_2, ..., m_n\}$ - множина всіх повідомлень, що формуються в стеганосистемі. Контейнер $c \in C$ – множина відкритих даних, яка використовується для вбудовування закритої інформації; $C = \{c_1, c_2, ..., c_q\}$ – множина всіх контейнерів, причому $q >> n$. Порожній контейнер – контейнер *c*, який не містить прихованої інформації. Заповнений контейнер – такий контейнер, який містить приховану інформацію (c_m). Заповнений контейнер не повинен візуально відрізнятися від порожнього. Контейнери бувають двох типів: потокові та фіксовані. Потокові контейнери – це послідовність бітів, яка постійно змінюється. Повідомлення вбудовується у нього в реальному режимі часу, тому заздалегідь невідомо, чи вистачить розміру даного контейнера для передачі повідомлення повністю [4]. Фіксовані ж контейнери мають фіксований розмір, тому є можливість обрати оптимальний контейнер для передачі повідомлення. Розмір контейнера повинен, принаймні, у декілька разів перевищувати розмір повідомлення, і, чим більше дане співвідношення, тим надійніше приховане повідомлення. Для підвищення рівня захищеності секретної інформації повідомлення можна попередньо зашифрувати стійким криптографічним алгоритмом. Також часто використовується завадостійке кодування. Типову схему стеганосистеми зображено на рис. 1.

5. Математична модель типової стеганосистеми

Процес звичайного стеганографічного перетворення описується такими залежностями:

$$E : C \times M \rightarrow S; \quad (1)$$

$$D : S \rightarrow M, \quad (2)$$

де

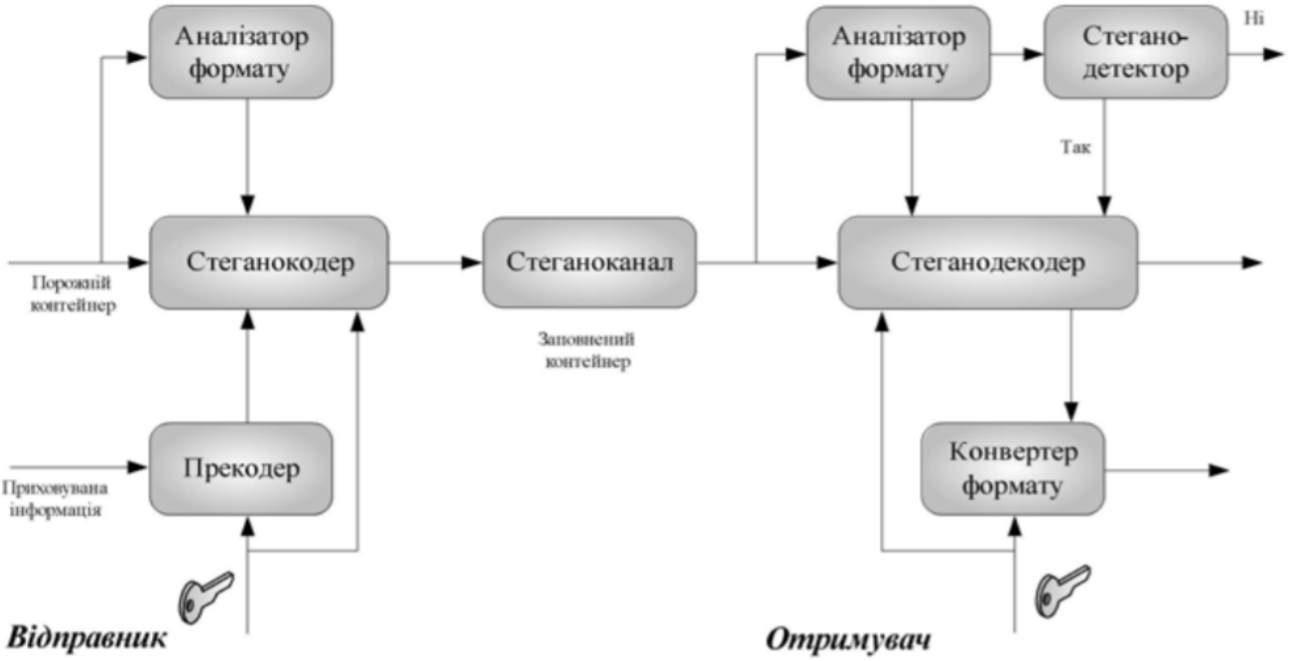


Рис. 1. Схема типової стеганосистеми

$S = \{(c_1, m_1), (c_2, m_2), ..., (c_q, m_q)\} = \{s_1, s_2, ..., s_q\}$ – множина заповнених контейнерів (стеганограм). Залежність (1) описує процес приховування інформації, залежність (2) – витягування прихованої інформації. Однією із обов'язкових умов при цьому є відсутність «перетину», тобто якщо $m_a \neq m_b$ (причому $m_a, m_b \in M$, а $(c_a, m_a), (c_b, m_b) \in S$), то $E(c_a, m_a) \cap E(c_b, m_b) = \emptyset$. В загальному випадку стеганосистему можна представити як сукупність ∑(C,M,S,E,D) – контейнерів, повідомлень та перетворень, що їх зв'язують. Завжди контейнери с обираються таким чином, щоб заповнений контейнер майже не відрізнявся від порожнього контейнера [4]. Стеганосистема може вважатися надійною, коли $sim[c, E(c,m)] = 1$ (де sim – функція подібності). Контейнер може обиратися двома способами: довільно (сурогатний метод) та підбором найбільш придатного у конкретному випадку контейнера, який зміниться найменше при перетворенні. В останньому випадку контейнер обирається виходячи із умови:

$$c = \max sim[x, E(x,m)]. \qquad (3)$$

В будь-якому випадку пряме та зворотне перетворення (E та D) мають відповідати одне одному та підлягати умові, що незначне викривлення контейнера (на величину δ) не має призводити до викривлення прихованої інформації:

$$E(c,m) \approx E(c + \delta, m) \text{ або}$$
$$D[E(c,m)] \approx D[E(c + \delta, m)] = m .$$

6. Модифікований алгоритм

Алгоритм 1. Алгоритм розподілу контейнера

Вхідні дані: секретне зображення I та секретний ключ

Вихід: n розподілів I1, I2, .., In

Крок 1. Перетворення секретного ключа у бінарний

Крок 2. Перетворення всіх пікселів (RGB) зображення I до бінарного

Крок 3. Вкраплення таємного ключа у синій колір за допомогою LSB

Крок 4. Множення або додавання перестановок за значенням ключа

Наприклад, розрахуємо значення (0,0) пікселів частин, що надсилатимуться до приймача. З рівняння маємо наступне:

$$S_{x=1}(0,0) = 161 + 61x \pmod{251} = 222$$
$$S_{x=2}(0,0) = 161 + 61x \pmod{251} = 32$$
$$S_{x=3}(0,0) = 161 + 61x \pmod{251} = 92$$
$$S_{x=4}(0,0) = 161 + 61x \pmod{251} = 154$$

Аналогічно розраховуються інші значення кольорів та пікселів.

Алгоритм 2. Шифрування AES

Вхідні дані: секретні дані

Вихід: зашифровані дані

Крок 1. AES шифрування з секретним ключем

Алгоритм 3. Відновлення секретного зображення та секретного ключа

Вхідні дані: будь-які зображення: I1, I2, ..., In

Вихід: секретне зображення та секретний ключ

Крок 1. Використати інтерполяцію Лагранжа I1, ..., Ik, щоб відновити сеганоконтейнер

Крок 2. Отримати стеганоконтейнер із зворотною перестановкою

Крок 3. Отримати секретне зображення та секретний ключ із дешифратором.

Алгоритм 4. Розшифровка AES

Вхідні дані: зашифровані дані

Вихід: секретні дані

Крок 1. AES розшифровка з секретним ключем

7. Результати дослідження

Вхідні дані:

- База даних із тисячею зображень
- Довільне секретне повідомлення для передачі
- Ключ для обрання бітів вкраплення

Результати:

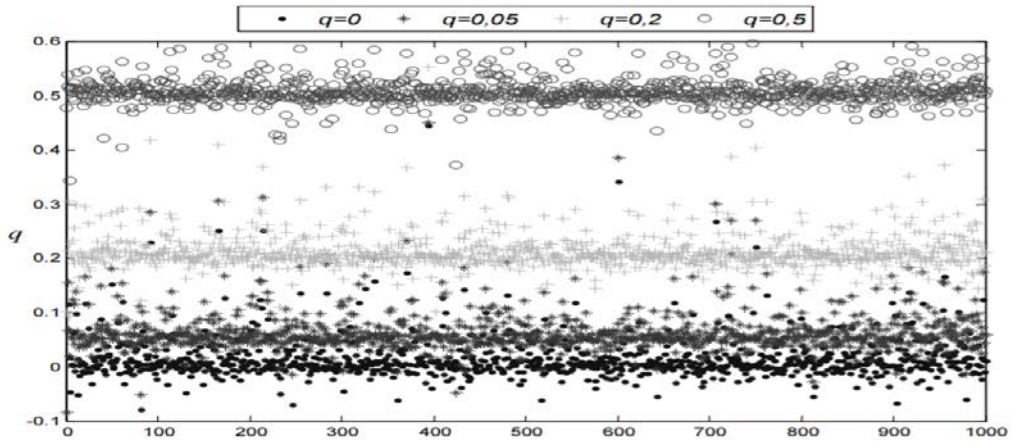


Рис. 2. Результати дослідження.

де q – частина зображень, які не "зламано" алгоритмом стеганоаналізу «Sample Pairs Analysis», в залежності від степені модифікації початкового алгоритму, а саме: порожній контейнер; стеганоконтейнер із вкрапленням методом LSB; стеганоконтейнер із вкрапленням методом LSB та накладанням додаткового ключа; стеганоконтейнер із вкрапленням повідомлення, попередньо розподіленого схемою Шаміра, методом LSB та накладанням додаткового ключа відповідно .

8. Висновки

В результаті дослідження, було продемонстровано взаємодію методів захисту інформації, направлену на підвищення безпеки передачі повідомлення та захисту його вмісту, а також показано результати стеганоаналізу на різних етапах алгоритму. Показано вагому перевагу методу найменш значущого біту в поєднанні зі схемою розподілу секрету Шаміра в порівнянні зі звичайним методом найменш значущого біту.

Стеганостійкість можна ще підвищити шляхом збільшення сторін розподілу секрету та сторін необхідних для відновлення, оскільки для відновлення початкового повідомлення використовується інтерполяція Лагранжа, цим самим підвищується ступіно многочлена, отже й зростає складність та ресурсозатратність у процесі відновлення.

Список літератури

1. Дж. Блеклі, 1979, "Захист криптографічних ключів", Proc. AFIPS 1979 National Computer Conference, 48, (1979), 313-317.
2. Тієн Ч. та Лін Я., 2006, "Розподіл секрету в зображеннях" Computers & Graphics, vol. 26, no. 5, 765–770.
3. Оуенс М., 2002, "Обговорення таємних каналів та стеганографії", SANS Institute.
4. Швидченко И.В. Методы стеганоанализа для графических файлов / И.В. Швидченко // Искусственный интеллект. – 2010. – № 4. – С. 697–705.

УДК 004.94

ТРЕГУБОВ А.В.,
ЛЯШКО С.І.

МАТЕМАТИЧНЕ МОДЕЛЮВАННЯ МАГНІТНОГО ПІДВІСУ У ТЕРМОЯДЕРНОМУ РЕАКТОРІ

Робота присвячена математичному моделюванні зон стійкості магнітного підвісу LDX. У якості запропонованого метода вирішення проблеми розглядаються метод з використанняями двох надпровідних кілець. Мета даної роботи полягає у підвищенні якості роботи термоядерного реактора за рахунок моделювання роботи магнітного підвісу та визначення зон стійкості в яких він працює оптимально та має максимальний коєфіцієнт корисної дії.

The work is devoted to the mathematical modeling of the stability zones of the magnetic suspension of LDX. As a proposed method of solution of the problem, a method is considered with the use of two superconducting rings. The purpose of this work is to improve the quality of the thermonuclear reactor by simulating the magnetic suspension and determining the zones of stability in which it operates optimally and has a maximum efficiency.

1. Вступ

Розвиток технічної цивілізації на Землі в XX ст. характеризується стрімким збільшенням енергоспоживання. За оцінками, в 1945— 1995 рр. населення планети використало 2/3 всього палива, добутого людством за час свого існування. Такі бурхливі темпи розвитку енергетики спричинили появу низки гострих проблем.

На перший план виходить проблема ресурсозабезпеченості енергетичного господарства. З одного боку, сумарні запаси паливних ресурсів досить великі, до того ж щороку стають відомими нові поклади викопного палива. Крім того, сучасна технологія відкриває доступ до використання нетрадиційних джерел енергії; це свідчить на користь того, що абсолютного дефіциту енергетичних ресурсів на планеті поки що не існує. З іншого боку, спостерігається відносна ресурсна обмеженість, зумовлена можливістю швидкого вичерпання найбільш доступних родовищ, і перехід до розробки складніших, що спричинює подорожчання енергоносіїв і робить використання більшої частини паливних ресурсів нерентабельним. Аналітики прогнозують наближення того моменту, коли енергетичні затрати на розвідування й добування головного виду палива — нафти — за межами Близького Сходу перевищуватимуть кількість енергії, яка може бути одержана з неї.

Методів вирішення її у вигляді технологій енергозбереження і використання поновлюваних джерел пропонується безліч, а однією з найперспективніших, але вельми віддалених від практичного впровадження заміною нинішнім електростанціям вважається термоядерний синтез. Експеримент, який відтворює схожі на земні магнітні поля, підтверджує потенціал нового способу створення реактора для вироблення енергії за допомогою злиття ядер - такий же реакції, яка відбувається на Сонці.

Керована реакція синтезу є жаданої мрією фізиків та інших дослідників вже понад півстоліття, тому як вона пропонує практично невичерпне джерело енергії без викидів сполук вуглецю і з набагато меншим радіоактивним забрудненням, ніж в заснованих на розподілі атомів АЕС. Однак побудувати реактор виявилося складніше, ніж вважалося спочатку. Просунути дослідження допоможуть нові результати від експериментальної установки в Массачусетському технологічному інституті (MIT), на розробку якої вчених надихнули космічні супутникові спостереження. У спільному проекті MIT і Колумбійського університету (Columbia University), LDX званому (Levitated Dipole Experiment - левітаційний дипольний експеримент),

використовується кільцеподібний магніт масою півтонни і розміром з покришку від великої вантажівки. Він виготовлений з розташованих усередині сталевої конструкції надпровідних котушок, утримується в "підвішеному" стані потужним електромагнітним полем і виконує функцію контролю за переміщенням зарядженого газу - плазми з температурою в 10 млн градусів, яка знаходиться у зовнішній камері діаметром 4,9 м.

2. Метод з використанням двох надпровідних кілець

Розглянемо систему, що складається з двох надпровідних кілець, у присутності однорідного поля ваги паралельного осі z. Одне з кілець зафіксовано таким чином, що його центр збігається з початком координат, а нормаль є спрямованою по осі z; друге кільце - вільне.

Потенційна енергія такої системи складається з магнітної енергії і енергії однорідного поля ваги:

$$V = \frac{1}{2}\frac{L_{22}\Psi_1^2 - 2\Psi_1\Psi_2 L_{12} + L_{11}\Psi_2^2}{L_{11}L_{22} - L_{12}^2} \pm G\cdot z \quad (1)$$

де L_1, L_2, L_{12}- власні і взаємні індуктивності; Ψ_1, Ψ_2- магнітні потоки; G- вага вільного кільця, а знак $''+''$ відповідає напрямку сили ваги, протилежному осі z; z- координата центра ваги вільного кільця, що збігається з його геометричним центром.

Через те, що досліджується можливість стійкої рівноваги такої системи, досить знати градієнт і гессіан від потенційної енергії системи у заданій точці. Основні труднощі пов'язані з обчисленням градієнта і гессіана від взаємної індуктивності такої системи. Для рішення цієї задачі проводять досить громіздкі викладення, в результаті чого повторний інтеграл удається звести до алгебраїчних комбінацій повних еліптичних інтегралів. Таким чином, в окремому випадку квазиспіввісного розташування кілець вдається одержати шукані вираження градієнта і гессіана потенційної енергії в спеціальних функціях. Проте, через надзвичайну громіздкість отриманих виражень подальше аналітичне дослідження достатніх умов рівноваги авторами не проводилося. На основі отриманих виражень, у разі конкретних значень параметрів, чисельно знаходили точку рівноваги і перевіряли умови стійкості.

Отримані аналітичні вираження придатні тільки для випадку квазиспіввісного розташування кілець, тобто вони не забезпечують достатнього апарату для дослідження довільної конфігурації кілець.

У цій главі пропонується альтернативний підхід до рішення зазначеної задачі, який заснований на чисельному моделюванні фізичних характеристик системи. Треба відмітити, що власне чисельні методи застосовувалися тільки до обчислення однократного інтеграла. Всі інші операції (наприклад, знаходження гессіанових підінтегральних функцій) виконували аналітично, з використанням символьних можливостей системи Maple V, у якій реалізовано механізм автоматичного диференціювання. Така послідовність обчислень забезпечує повний контроль точності виражень, що обчислюють, у тому числі градієнтів і гессіанов. Ще однією перевагою даного підходу є універсальність, тобто його застосовують для дослідження багатокільцевих конфігурацій з довільним розташуванням кілець. У цій главі приведено базову процедуру такого підходу, а саме, підінтегральну функцію взаємної індуктивності.

Потенційна енергія системи, її градієнти і гессіани були приведені до безрозмірного виду за допомогою безрозмірних змінних такого виду :

$$\overline{m} = L_{12}\Big/\sqrt{L_1 L_2} \quad (2)$$

$$p = \sqrt{\Psi_1^2 L_2\Big/\left(\Psi_2^2 L_1\right)} \quad (3)$$

де $\overline{m}, p$- відповідно безрозмірна взаємна індуктивність і параметр, що характеризує відношення магнітних потоків.

Недоліком такого вибору безрозмірних параметрів є залежність безрозмірної взаємної індуктивності від магнітних параметрів, тоді як насправді її цілком визначають геометричні параметри. Через те, що основні труднощі обчислення пов'язані саме з обчисленням взаємної індуктивності, її градієнта і гессіана, необхідно прагнути до мінімального набору

аргументів у цих функціях. Тому, як безрозмірні параметри, зручно вибрати наступні величини:

$$l^2 = \kappa_0 \overline{L}_1 \overline{L}_2 \quad (4)$$

$$q = \overline{L}_2 \Psi_1 / \Psi_2 \quad (5)$$

$$\overline{L}_i = 4\pi L_i / (\mu_0 x_i) \quad (6)$$

$$4\pi L_{12}(x_0, y_0, \vec{R}_0, \theta, \varphi) / (\mu_0 y_0) = m(\kappa_0, \vec{r}_0, \theta, \varphi) = m \quad (7)$$

де $\kappa_0 = x_0 / y_0$ - відношення радіусів кілець; $\overline{L}_i$, $(i = 1,2)$ - власна індуктивність у безрозмірній формі; Ψ_i, $(i = 1,2)$ - "заморожені" у кільцях магнітні потоки; $x_i, (i = 1,2) \rightarrow (x_0, y_0)$ - радіуси кілець; L_{12} - взаємна індуктивність двох кілець; $\vec{R}_0, \theta, \varphi$ - координати, що задають просторове положення вільного кільця. Безрозмірна енергія має вид:

$$u = 2 \cdot (L_2 V) / \Psi_2^2 = \frac{q^2 - 2mq + l^2}{l^2 - m^2} \quad (8)$$

Встановимо відповідність між безрозмірними параметрами:

$$p = q / \sqrt{l} \quad (9)$$

$$\overline{m} = m / \sqrt{l} \quad (10)$$

$$\delta = \kappa_0 \quad (11)$$

Крім згаданих помилок у вираженнях компонентів гессіана взаємної індуктивності, в зазначеній роботі є суперечне твердження: "стійка рівновага можлива лише в системі магнітно-притягувальних кілець однакових радіусів". Точніше, сумнів викликає твердження про необхідність близькості радіусів кілець для забезпечення стійкості в конфігурації "магнітного підвісу".

Досліджуємо необхідні і достатні умови магнітної левітації.

Істинний мінімум потенційної енергії можливий тільки на осі z, тому що в противному випадку в точці рівноваги градієнт і гессіан магнітної енергії будуть залежати від φ, а, як відомо, зміна φ не приводить до зміни відповідних компонентів гессіана, тобто існує не одна, а нескінченна кількість точок рівноваги, які відповідають різним $\varphi \in [0, 2\pi)$; це також випливає з розумінь симетрії. У будь-якій точці осі залежність від φ вироджується, і тому необхідно аналізувати позитивну визначеність урізаної матриці гессіана розміром 4×4, а не 5×5.

Для знаходження зон стійкої рівноваги і точок магнітної левітації скористаємося наступною процедурою:

- шукаємо одномірний мінімум магнітної енергії по осі z, щоб переконатися в тому, що при заданих параметрах магнітна енергія має мінімум у точці, відмінній від центра нерухомого кільця;

- знаходимо максимум і мінімум градієнта магнітної енергії по осі z, щоб визначити точку z_1, яка є лівою границею інтервалу з передбачуваним нулем сили;

- на інтервалі $[z_1, \infty[$ знаходимо точку z_2, у якій магнітна сила дорівнює нулю;

- на інтервалі $[z_2, \infty[$ шукаємо точку максимуму сили (z_{max}), у якій

3-я умова системи

$$\begin{cases} \dfrac{\partial^2 u}{(\partial x)^2} > 0; \quad \dfrac{\partial^2 u}{(\partial y)^2} > 0; \quad \dfrac{\partial^2 u}{(\partial z)^2} > 0; \\ \dfrac{\partial^2 u}{(\partial x)^2} \dfrac{\partial^2 u}{(\partial \theta)^2} > \dfrac{\partial^2 u}{\partial x \partial \theta}. \end{cases} \quad (12)$$

вже не виконується;

- для визначення лівої границі діапазону стійкості z_{min} розв'язуємо рівняння:

$$\frac{\partial^2 m}{(\partial x)^2} \frac{\partial^2 m}{(\partial \theta)^2} - \left(\frac{\partial^2 m}{\partial x \partial \theta} \right)^2 = 0 \quad (13).$$

Рівняння (13) виводять з 3-ї умови системи (12) з урахуванням формул.

- будуємо дві криві, що обмежують діапазон стійкості по z, а саме, компоненти гессіана $\partial^2 u / (\partial x)^2$ і $\partial^2 u / (\partial z)^2$ як функції z (позначені 1 і 2 відповідно);

- область, що обмежена кривими 1 і 2, знайденими в попередньому пункті і яка попадає до діапазону $]z_{min}, z_{max}[$, і є областю стійкості;

- нуль магнітної сили z_2 (ефект МПЯ Козоріза), як правило, знаходиться на інтервалі $]z_{min}, z_{max}[$, але ніколи не попадає до області, яка обмежена кривими 1 і 2.

- для забезпечення достатніх умов стійкості точку рівноваги необхідно змістити

в область, обмежену кривими 1 і 2. Гравітаційну силу, наявність якої не змінює достатні умови існування мінімуму потенційної енергії, але змінює необхідну умову рівноваги, використовують для досягнення цієї мети. Варіюючи масою підвісу, можна змістити точку нуля сили в область стійкості.

3. Результати чисельних екстпериментів

Характерні результати чисельних експериментів по знаходженню областей стійкості представлено на рис 1 за різних значень параметрів δ і p.

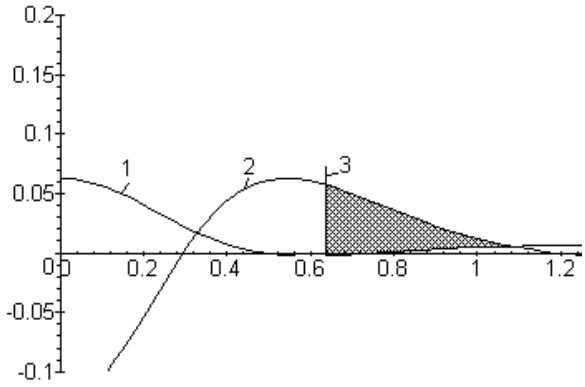


Рис. 1. Залежність області стійкості від δ

На рис.1 побудовано криві 1-3 за значень $\delta = 0.1$, $p = 0.1$.

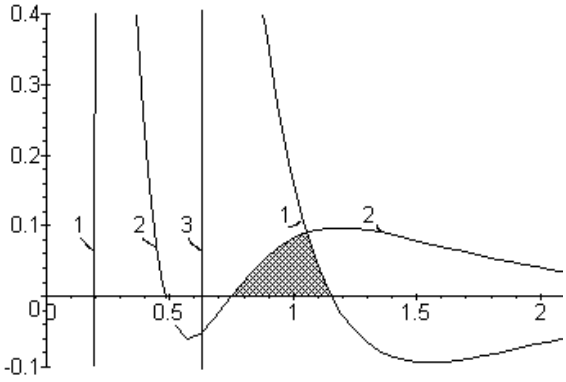


Рис. 2. Залежність області стійкості від p

На рис.2 побудовано криві 1-3 за значень $\delta = 0.2$, $p = 0.4$.

Аналіз результатів чисельного дослідження показує, що в розглянутій системі є можливою магнітна левітація у вигляді підвісу. Величину області стійкості визначають двома безрозмірними параметрами системи δ і p. Характер залежності величини області стійкості від зміни параметра δ показано на рис.1, а вплив параметра p на величину області стійкості - на рис.2.

9. Висновки

У даній роботі було розглянуто моделювання зон стійкості магнітного підвісу LDX за допомогою метода з використанням двох надпровідних кілець. Також було створено математичну модель та проведено чисельні експерименти, за результатами якої виявилося, що найбільша магнітна стійкість спостерігається для параметра $\delta = 0.2$

Список літератури

1. Уайт Давид С., Вудсон Герберт Х. “Электромеханическое преобразование энергии”. -Москва; Ленинград: Энергия, 1964. –528 с.
2. Миткевич В.Ф. Магнитный поток и его преобразования. -Москва; Ленинград: Изд-во АН СССР, 1946. – 358 с.
3. Magnetic Levitation and MHD Propulsion, P.Tixador, J.Phys. III France 4, 1994. –C.581-593
4. Козорез В.В. Динамические системы магнитно взаимодействующих Свободных тел. -Киев: Наукова думка, 1981.-140 с.
5. Смайт В. Электростатистика и электродинамика. -Москва; Ленинград: Гостехиздат, 1954. – 604 с.

УДК 334.021

ТРОЦЮК А.Р.
ПОПЕНКО В.Д.

ДОСЛІДЖЕННЯ СТРАТЕГІЙ ПОВЕДІНКИ ЗЕМЛЕКОРИСТУВАЧІВ У СІЛЬСЬКОГОСПОДАРСЬКОМУ ВИРОБНИЦТВІ ЗАЛЕЖНО ВІД ФОРМ ВЛАСНОСТІ НА ЗЕМЛЮ

У статті розглянутий підхід до обґрунтування необхідності впровадження ринку землі на основі повноцінної власності на неї на противагу пануючим сьогодні відносинам оренди. Обґрунтована більш раціональна поведінка землевласника, що передбачає збереження родючості землі в довгостроковому плані, на відміну від поведінки орендаря, орієнтованого на отримання максимального прибутку за період оренди земельної ділянки.

The article considers the approach to substantiating the necessity of introducing a land market on the basis of land full ownership, in contrast to the current lease relations. More rational behavior of the landowner is justified, which implies preserving land productivity in the long run, in contrast to the tenant's behavior, focused on obtaining maximum profits for the land lease period.

1. Вступ

Питання впровадження в Україні повноцінного ринку землі є сьогодні предметом гострих політичних дискусій як серед народних депутатів, так і серед звичайних громадян. В той же час відчувається брак економічного обґрунтування того чи іншого рішення, підкріпленого математичним розрахунком.

На цьому шляху будемо відштовхуватись від явища виснаження землі, що проявляється як падіння її урожайності внаслідок тривалого вирощування на ній тої чи іншої прибуткової монокультури. Така поведінка характерна для орендаря земельного масиву, який по закінченню терміну оренди залишає землю власнику-орендодавцю в виснаженому стані. На відміну від нього власник земельного масиву не обмежений в експлуатації землі терміном оренди і тому слід очікувати від нього дотримання циклів сівозміни, які підтримують продуктивність землі в сталому стані за рахунок чергування культур рослинництва. Таким чином, раціональну поведінку власника землі визначає компроміс між прибутковістю (і, відповідно, привабливістю) культур рослинництва і необхідністю дотримання циклів сівозміни. Приблизно те саме можна сказати щодо протиерозійних заходів: у них зацікавлений власник землі, на відміну від орендатора.

Проблема деградації ґрунтів набула всесвітнього масштабу. Це викликало стурбованість міжнародної спільноти і укладення Конвенцію ООН для боротьби з опустелюванням (UNCCD), яку ратифікували майже 200 країн світу [1]. В той же час вплив форм власності на землю на проблему деградації ґрунтів виглядає як недостатньо усвідомлений.

Зважаючи на різні стратегії поведінки землевласника та орендаря стосовно використання землі, можна поставити окремі задачі. Адаптуємо постановку задачі з [2] для цих двох випадків. Причому сформулюємо її у більш зрозумілій формі, зокрема не будемо розглядати кілька полів, а натомість залишимо одне поле і будемо лише його розглядати з точки зору планування сівозміни на певний проміжок часу. Це ніяк не вплине на висновок. Також будемо вважати, що початкові умови для ведення господарства у землевласника та орендаря однакові, тобто стан поля є

придатним для використання і не потребує додаткової обробки.

2. Задача оптимізації сівозміни для землевласника

Вважаємо, що власник земельної ділянки оптимізує один цикл сівозміни, але враховує, що в останньому році сівозміни має бути посіяна культура, прийнятна як попередник для першої в циклі культури. Дохід від земельної ділянки за період T можна розрахувати за формулою:

$$\left(\sum_{t=2}^{T} p(c_{t-1}, c_t)\right) + p(c_T, c_1),$$

де t – номер року в періоді сівозміни, T – період сівозміни в роках, c_t – культура, що вирощується на t-му році, c_{t-1} – культура, що вирощується в $t-1$-й рік, тобто попередник культури c_t, $p(c_{t-1}, c_t)$ – прибуток з 1 га культури c_t, що вирощується після культури c_{t-1}.

Цільова функція:

$$max\left(\sum_{t=2}^{T} p(c_{t-1}, c_t)\right) + p(c_T, c_1),$$

$$c_{t-1} \in F(c_t) \forall t \in [1, T]$$

$$c_0 = c_T,$$

де $F(c_t)$ – множина можливих попередників культури c_t.

3. Задача оптимізації сівозміни для орендаря

Вважаємо, що власник земельної ділянки оптимізує один цикл сівозміни, але враховує, що в останньому році сівозміни має бути посіяна культура, прийнятна як попередник для першої в циклі культури. Прибуток за період T можна розрахувати за формулою:

$$\sum_{t=1}^{S} p(c_{t-1}, c_t)$$

де t – номер року в періоді сівозміни, S – період оренди в роках, c_t – культура, що вирощується на t-му році, c_{t-1} – культура, що вирощується в $t-1$-й рік, тобто попередник культури c_t, $p(c_{t-1}, c_t)$ – прибуток з 1 га культури c_t, що вирощується після культури c_{t-1}, $p(c_0, c_t)$ – прибуток з 1 га культури c_t без попередника (на ідеальному полі).

Цільова функція:

$$max \sum_{t=1}^{S} p(c_{t-1}, c_t),$$

$$c_{t-1} \in F(c_t) \forall t \in [1, S].$$

При цьому завжди $c_0 \in F(c_t) \forall t \in [1, S]$

Дану задачу можна вирішувати методом динамічного програмування. Тому побудуємо рекурентну формулу:

$$q(c_t) = \max_{c_{t-1} \in F(c_t)} \left(q(c_{t-1}) + p(c_{t-1}, c_t)\right)$$

де $q(c_t)$ — сума прибутків з 1-го по t-й рік. Для застосування формули потрібно покласти $q(c_0)$ рівним нулю та продовжити розрахунки до $t = S$.

4. Врахування структури існуючих даних

Почнемо із формування функції $p(c_{t-1}, c_t)$. Інформація про зв'язок культури-попередника та культури-наступника є в [3] ст. 170, вона подана у вигляді матриці, де рядок відповідає культурі-попереднику, а стовпець – культурі-наступнику. На перетині цих культур міститься позначення: Х – хороший; Д – допустимий; Н – недопустимий. Ці позначення можна трансформовані в коефіцієнти для зручності обрахунків. Зміст цього коефіцієнту – це частка прибутку у відсотках від чистого прибутку на ідеальному полі, якого буде досягнуто при використанні певної пари попередник-наступник. Вважається, що відмова від трипілля і перехід до сівозмін дозволили у два рази підвищити урожайність [3] ст. 142. Тому позначення можна трансформувати так:

- Х – 100%
- Д – 75%
- Н – 50%

Таким чином, функцію $p(c_{t-1}, c_t)$ можна визначити як добуток вищезгаданої частки на чистий прибуток ідеального поля з 1 га, за який можна прийняти прибуток розрахований із даних [4], [5].

Оскільки навіть для недопустимих попередників функція $p(c_{t-1}, c_t)$ повертатиме певний результат, то відпадає необхідність у

використанні функції $F(c_t)$. Тому задачі матимуть вигляд:

- для землевласника

$$max\left(\sum_{t=2}^{T} p(c_{t-1}, c_t)\right) + p(c_T, c_1)$$

- для орендаря

$$max\sum_{t=1}^{S} p(c_{t-1}, c_t)$$

із рекурентною формулою для розв'язання

$$q(c_t^i) = \max_j \left(q(c_{t-1}^j) + p(c_{t-1}^j, c_t^i)\right),$$

де c_t^i – i-та культура, яка могла б використовуватись в рік t.

5. План досліджень

Оскільки план дій орендаря враховує менше обмежень, ніж план дій власника, за той же період земельна ділянка принесе орендарю більший дохід, ніж власнику. Але він залишить ділянку в стані, в якому необхідне її відновлення для початку нового циклу. Це, як очікується, підтвердить гіпотезу щодо практики оренди землі як суттєвого фактору її виснаження.

Перелік посилань

1. К.Ситник, В.Багнюк. Стан ґрунтів і майбутнє людства. //Вісник НАН України, 2008, № 8
2. Светлов Н.М., Модели динамического программирования для оптимизации севооборотов / Проблемы формирования аграрного рынка России / Н.М Светлов - М.: Изд-во МСХА, 1997. – С. 467-471.
3. В. П. Гудзь, І. Д. Примак, Ю. В. Будьонний, С. П. Танчик ЗЕМЛЕРОБСТВО.
4. Сільське господарство України. Статистичний збірник. //Держстат України, 1998-2018. [Електронний ресурс] // Режим доступу: http://www.ukrstat.gov.ua/druk/publicat/Arhiv_u/07/ /Arch_sg_zb.htm
5. Витрати на виробництво продукції сільського господарства в сільськогосподарських підприємствах. Статистичний бюлетень //Держстат України, 1998-2018. [Електронний ресурс] // Режим доступу: http://www.ukrstat.gov.ua/druk/publicat/Arhiv_u/07/ Arch_%D0%BEehv_bl.htm

УДК 004.942

ШУЛЬКЕВИЧ Т.В.
СЕЛІН Ю.М.

МАТЕМАТИЧНИЙ АПАРАТ ІНТЕЛЕКТУАЛЬНОГО АНАЛІЗУ ДАНИХ ДЛЯ ПРОГНОЗУВАННЯ НЕЛІНІЙНИХ НЕСТАЦІОНАРНИХ ПРОЦЕСІВ. ОПИС ТА РЕЗУЛЬТАТИ ТЕСТУВАННЯ

Описано математичний апарат, що можна застосовувати в задачах прогнозування поведінки нелінійних нестаціонарних процесів різної природи. Викладено інформаційну технологію для прогнозування нелінійних нестаціонарних процесів. Наведено результати застосування інформаційної технології для прогнозування таких процесів.

A mathematical apparatus, that can be used in problems of prediction of behavior of nonlinear non-stationary processes of different nature is described. The information technology for prediction of nonlinear non-stationary processes is presented. The results of application of information technology for forecasting of such processes are resulted.

Вступ

Інтенсивний розвиток науки призвів до появи великої кількості методів прогнозування поведінки процесів різної природи що отримуються у вигляді часових рядів [1,2]. Зазначимо, що майже всі вони є нелінійними та нестаціонарними (можна казати виключно про кусочну лінійність та кусочну стаціонарність).

Але вся ця кількість наявних методів прогнозування не дають гарантію що вони покривають вся можливі варіанти розвитку ситуації. Разом з тим, переважна більшість методів є двохетапними. На першому етапі аналізуються параметри раду, що прогнозується, на другому етапі обирається відповідний метод прогнозування і, нарешті, отримується прогноз. Але що робити, коли параметри ряду змінюються? Один метод вже неможна використовувати, бо параметри ряду вже змінилися, а обрати інший ще неможна, бо процес зміни параметрів ще не завершився.

Тож виникає необхідність розробки математичного апарату задля створення нових методів аналізу та прогнозування нелінійних нестаціонарних процесів різної природи з метою підвищення адекватності математичних моделей нелінійних нестаціонарних процесів та покращення якості оцінок прогнозів, які обчислюються за допомогою побудованих моделей. Розробку будемо провадити із застосуванням інтелектуальних методів аналізу даних.

Інтелектуальні методи аналізу даних за визначенням - це процес пошуку у «сирих» даних раніше не відомих, нетривіальних, практично корисних і доступних інтерпретацій знань, необхідних для прийняття рішень у різних сферах людської діяльності [3]. Інтелектуальний аналіз даних (ІАД) - термін, що застосовується для опису здобуття знань у базах даних, дослідження даних, обробки зразків даних, очищення і збору даних. Це процес виявлення кореляції, тенденцій, шаблонів, зв’язків і категорій [4] Інтелектуальний аналіз даних розвивається на базі таких наук, як прикладна статистика, розпізнавання образів, штучний інтелект, теорія баз даних тощо.

Процес автоматичного пошуку прихованих закономірностей або взаємо’звязків між змінними в інтелектуальному аналізі даних поділяється на задачі класифікації, моделювання і прогнозування з використанням статистичних та математичних методів.

Складні задачі, на макрорівні, характеризуються аналізом та прогнозуванням нелінійних нестаціонарних процесів. У зв’язку із цим пропонуються до розгляду деякі методи, що можна застосовувати для використання

в подібних задачах. До них можна віднести наступні методи:

- приховані марковські моделі;
- лінгвістичне моделювання.

Метод прихованих марковськіх моделей.

Традиційно приховані марковські моделі (ПММ) визначають як трійку

$\lambda = (A, B, \Pi)$,

де 1) $A = \{a_{ij}\}$ - матриця ймовірностей переходів зі стану S_j в стан S_i ,

$a_{ij} = P\lfloor q_{i+1} = S_j \mid q_i = S_i \rfloor, 1 \le i, j \le N$;

2) $B - \{b_j(k)\}$ - розподіл ймовірностей спостережуваних символів в стані j ? де $b_j(k) = P\lfloor v_k \mid q_i = S_j \rfloor, \quad 1 \le j \le N, \quad 1 \le k \le M$ (для безперервного випадку $b_j(k)$ задається як функція розподіли щільності ймовірності).

3) $\Pi = \{\Pi_i\}$ - ймовірність кожного початкового стану,

Загальну схему функціонування прихованої марковської моделі зображено на рис.1.

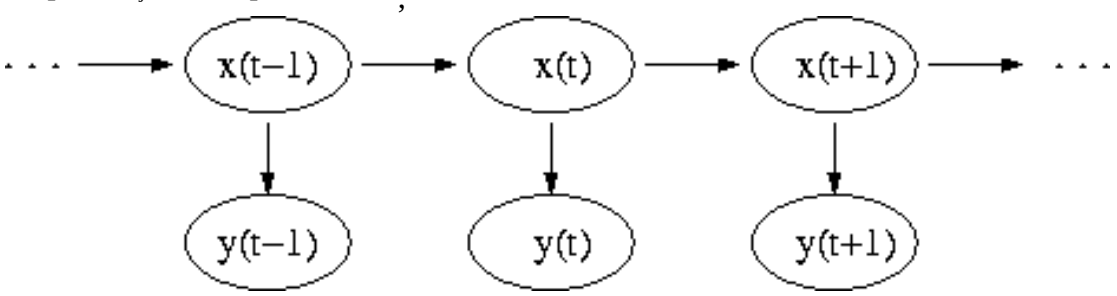


Рис 1. Загальна схема функціонування прихованої марковської моделі.

Основні завдання при застосуванні ПММ до визначення параметрів процесів. Для використання ПММ при розпізнаванні мови необхідно вирішити три задачі [5].

Завдання 1: Якщо задані послідовність спостережень і модель $\lambda = (A, B, \Pi)$ то як ефективно обчислити $P(O \mid \lambda)$ - вірогідність такої послідовності при заданих параметрах моделі?

Завдання 2: Якщо задані послідовність спостережень і модель $\lambda = (A, B, \Pi)$ то як визначити відповідну послідовність внутрішніх станів ?

Завдання 3: Якщо задані послідовність спостережень, то як визначити параметри моделі $\lambda = (A, B, \Pi)$, виходячи з критерію максимізації $P(O \mid \lambda)$?

Кожна з цих трьох задач має відповідний алгоритм вирішення. Перша задача – алгоритм «вперед-назад», друга – алгоритм Вітербі, третя – Баума-Велша.

Розв'язання першої задачі дозволяє розраховувати ймовірність що певна полідовність була оримана за допомогою відповідної моделі. А, за допомогою знайденої моделі можна зробити прогноз.

Лінгвістичне моделювання

Згідно з етапів побудови лінгвістичної моделі(ЛМ) вихідна задача буде розбита на такі підзадачі:

підзадача отримання різницевих рядів;
підзадача інтервалізації;
підзадача лінгвістизації;
підзадача побудови матриці переходів.

Отриману послідовність символів аналізують на наявність граматичних конструкцій. На виході отримуємо список граматичних конструкцій з ймовірностями їх наявності в процесі, а також матрицю ймовірностей переходу з символу в символ.

Підзадача отримання різницевих рядів. Призначенням даної підзадачі є отримання рядів, котрі характеризують динаміку зміни руху курсору «мишки»: швидкість(різницевий ряд 1-ого порядку), прискорення(різницевий ряд 2-ого порядку) і т. д. Таким чином різницеві ряди являються похідними від вихідного ряду.

Дано: Вектор цілих чисел $\bar{X}$ з потужністю $n = |\bar{X}|$.

Результати: Вектор цілих чисел $\overline{D}$ з потужністю $k = |\overline{D}|$.

Обмеження:

$$\forall d_i \in \overline{D} : d_i = x_{i+1} - x_i \quad (1)$$

де $i \in [0; n-1); x_{i+1}, x_i \in \overline{X}$

$$k = n - 1 \quad (2)$$

Підзадача інтервалізації. Призначенням даної підзадачі є побудова алфавіту користувача шляхом розбиття відсортованого різницевого ряду на множину інтервалів, кожний елемент якої характеризує певну літеру алфавіту.

Дано:

– гіпотетична потужність алфавіту a;

– вектор цілих чисел $\overline{D}$ з потужністю $k = |\overline{D}|$.

Результати:

Вектор пар цілих значень $\overline{I}$ з потужністю $n = |\overline{I}|$.

Обмеження:

$$\forall x \in \overline{I} : x^1 \le x^2 \quad (3)$$

$$\forall x_i, x_{i+1} \in \overline{I} : x_i^2 < x_{i+1}^1, \quad (4)$$

де $i \in [0; n-1)$

$$n \le a \quad (5)$$

$$a \ll k \quad (6)$$

$$\exists x \in \overline{I} : \forall d \in \overline{D}, d \in [x^1; x^2] \quad (7)$$

$$\forall d_i, d_{i+1} \in \overline{D} : d_i \le d_{i+1}, \quad (8)$$

де $i \in [0; k-1)$

$$x_0 \in \overline{I} : x_0 = (-\infty; x_1^1) \quad (9)$$

$$x_n \in \overline{I} : x_n = (x_{n-1}^2; +\infty) \quad (10)$$

Підзадача лінгвістизації. Призначенням даної підзадачі є отримання лінгвістичного ланцюжку шляхом знаходження відповідної літери алфавіту для кожного значення різницевого ряду. Літера алфавіту однозначно відповідає певному інтервалу з множини інтервалів, отриманих в результаті розв'язання попередньої задачі.

Дано:

– вектор цілих чисел $\overline{D}$ з потужністю $k = |\overline{D}|$, що відповідає обмеженню представленому у формулі 7;

– вектор пар цілих значень $\overline{I}$ з потужністю $n = |\overline{I}|$ з обмеженнями, що наведені у формулах 1.3 та 1.4, а також у 9 та 10.

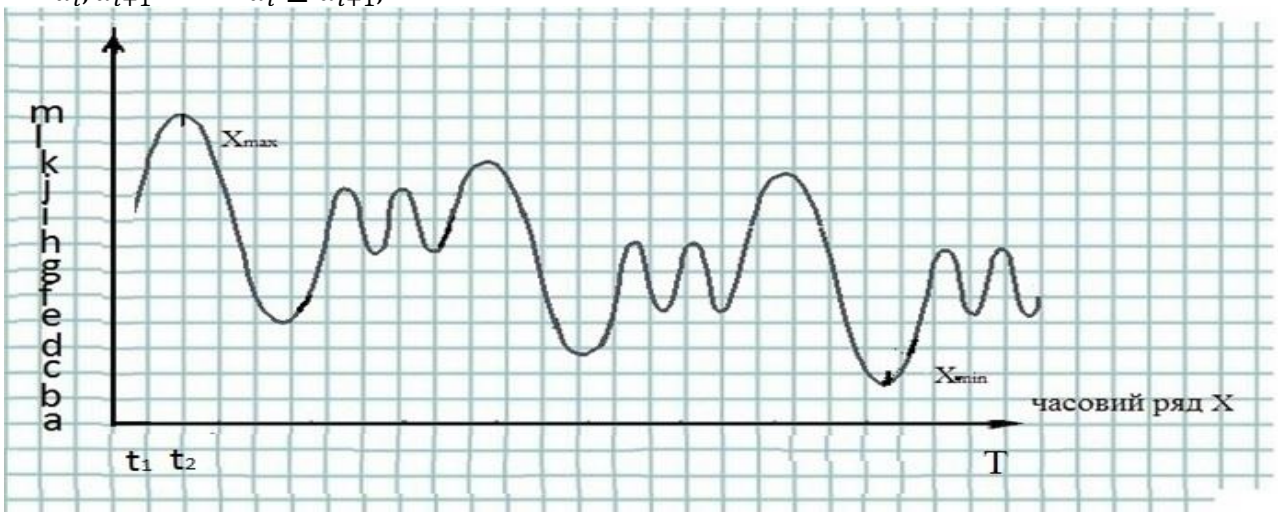
Результати:

Вектор цілих чисел $\overline{A}$ з потужністю k.

Обмеження:

$$\forall x_i \in \overline{A} : \exists d_i \in \overline{D}, \exists y_j \in \overline{I}, d_i \in [y_j^1; y_j^2], x_i = j, \quad (11)$$

де $i \in [0; k), j \in [0; n)$



$F(t_1) \sim k$

$F(t_2) \sim m$

k m k f e h i j h k k g d e g d h k k g d e g d h

Рис.2 Загальна схема переходу до лінгвістичного ланцюжку

За допомогою викладеного математичного апаратута розроблених алгоритмів було проаналізовано кілька часових рядів (вартість акціц Swiss-air, індекс D-J, вартість золота).

За максимальний розмір файлу бралось 4000 значень часового ряду з поступовим зменшенням розміру ряду до 200 з кроком 200.

Результати розрахунків наведено на рис.3.

Розрахунки повторювались на різних розмірностях вхідного ряду для отриманих рядів без тренду (різницевих рядів). На рис. 4 показано отримані результати.

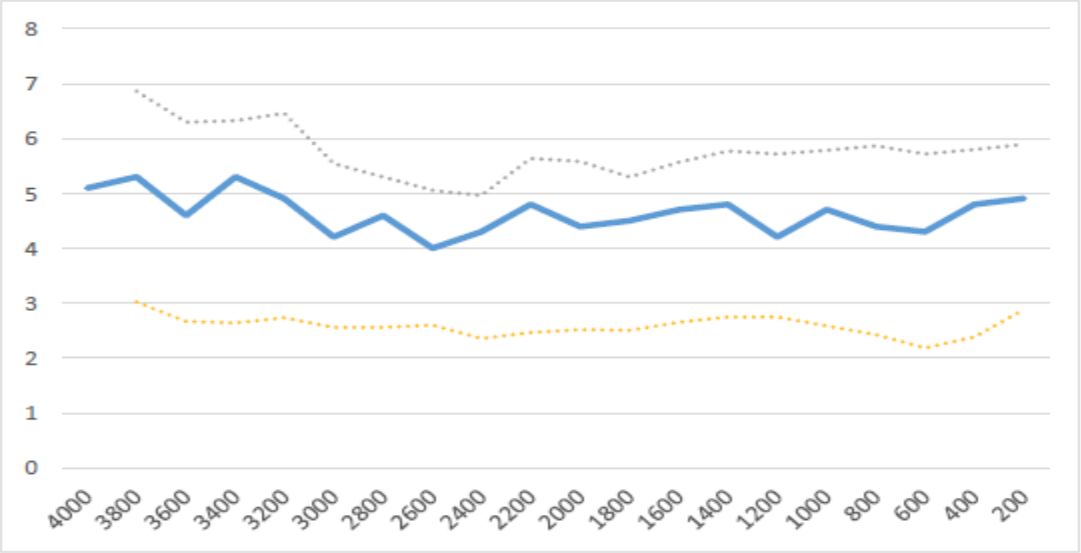


Рис. 3. Зміна кількості успішних прогнозів тренду для різної розмірності вхідного ряду.

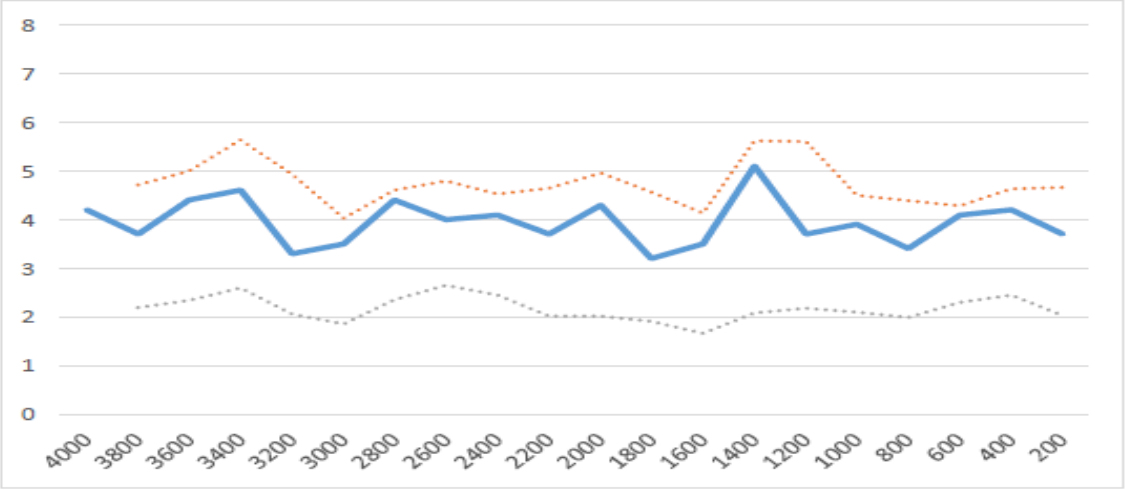


Рис. 4. Зміна кількості успішних прогнозів тренду для початкового ряду позбавленого тренду (різницевого ряду).

Висновки

Таким чином було розроблено математичний апарат для обчислення кількісних та якісних показників часових рядів економічної та екологічної природи з використанням прихованих марковських моделей та лінгвістичного моделювання, який відрізняється стійкістю отримуваних результатів і забезпечує підвищення якості прогнозів відповідних показників.

Сформовано структури нових математичних моделей нелінійних нестаціонарних процесів, які відрізняються спрощенням процедури побудови моделі і забезпечують опис високого рівня адекватності для досліджуваних процесів.

Розроблено метод обробки екстремальних значень, які відрізняється незначними обчислювальними витратами і забезпечує отримання однорідних вибірок даних та можливість їх використання для побудови моделей.

Список літератури

1. Бідюк П.І. Аналіз часових рядів. [Текст] Навчальний посібник / Бідюк П.І., Романенко В.Д., Тимощук О.Л. / – К.: НТУУ «КПІ», 2013. – 600 с.

2. Douglas C. Montgomery, Cheryl L. Jennings, Murat Kulahci. Introduction to Time Series Analysis and Forecasting, 2nd Edition, 2015, 672pp.

3. Fayyad U. M. From Data Mining to KnowledgeDiscovery in Databases / U. Fayyad, G. Piatetsky-Shapiro, P. Smyth // AI Magazine. – 1996. – №17(3). – P. 37–54

4. Плескач В. Л. Інформаційні системи і технології на підприємствах : підручник / В. Л. Плескач, Т. Г. Затонацька. - К. : Знання, 2011. - 718 с.

5. Баклан І. В. Лінгвістичне моделювання: основи, методи, деякі прикладні аспекти. Систем. технології. — 2011. — № 3. — С. 10-19.

6. Баклан І.В. Структурний підхід до аналізу та прогнозування поведінки часових рядів / І.В. Баклан, Ю.М. Селін // Вісник Херсонського національного технічного університету. — Херсон: ХНТУ, 2005. — № 2. — С.27–31.

УДК 519.854.2

ГОДНА А.В.
ЖДАНОВА О.Г.
СПЕРКАЧ М.О.

ЗАДАЧА МІНІМІЗАЦІЇ СУМАРНОГО ВІДХИЛЕННЯ МОМЕНТІВ ЗАВЕРШЕННЯ ВІД ДИРЕКТИВНИХ СТРОКІВ ПРИ ВИКОНАННІ ЗАВДАНЬ ПАРАЛЕЛЬНИМИ ПРИСТРОЯМИ

У роботі розглянуто задачу теорії розкладів, у якій необхідно скласти розклад виконання завдань паралельними пристроями за критерієм мінімізації сумарного відхилення моментів завершення від директивних строків. Особливістю задачі є те, шо є дві множини завдань, кожна зі своїм директивним строком. Розроблено евристичний алгоритм складання розкладу, що складається з трьох етапів. Сформульовані критерії розподілу множини пристроїв за множинами завдань та відповідна допоміжна оптимізаційна задача. Розроблено алгоритм побудови допустимого розкладу. Запропоновано множину перестановок завдань, які призводять до покращення допустимого розкладу. Проведено серію експериментів для порівняльного аналізу запропонованих критеріїв розподілу пристроїв за множинами завдань.

The article presents the problem of scheduling theory, in which it is necessary to schedule the tasks by parallel machines on the criterion of minimizing the total deviation of the completion of tasks from the due dates. A special feature of the problem is that there are two sets of tasks, each with its own due date. A heuristic algorithm for scheduling, consisting of three stages, has been developed. The criteria for assigning devices to sets of tasks and the corresponding auxiliary optimization problem are formulated. An algorithm for constructing an acceptable schedule has been developed. The set of permutations of tasks, which lead to improvement of the allowable schedule, is proposed. A series of experiments was conducted for the comparative analysis of the proposed criteria for the distribution of devices in sets of tasks.

Ключові слова: РОЗКЛАД, ДИРЕКТИВНИЙ СТРОК, ПАРАЛЕЛЬНІ ПРИСТРОЇ, ІДЕНТИЧНІ ПРИСТРОЇ, ПРОПОРЦІЙНІ ПРИСТРОЇ, ЗАПІЗНЕННЯ, ВИПЕРЕДЖЕННЯ, МІНІМІЗАЦІЯ СУМАРНОГО ВІДХИЛЕННЯ.

1. Постановка задачі

Задано дві множини завдань $I_1 = \{1, 2, ..., n_1\}$ та $I_2 = \{1, 2, ..., n_2\}$ та кількість пристроїв $m > 1$. Множина завдань I_1 має спільний директивний строк d_1, а множина I_2 – директивний строк d_2. Пристрої працюють паралельно, є взаємозамінними і відрізняються один від одного продуктивністю виконання завдань. Пристрої можна впорядкувати за швидкістю виконання завдання і цей порядок однаковий для всіх завдань. Для кожного пристрою j, $j = \overline{1, m}$ задано коефіцієнт продуктивності h_j такий, що тривалість виконання завдання i на пристрої j дорівнює $h_j p_i$. «Еталонним» будемо називати пристрій з коефіцієнтом продуктивності $h = 1$. У цьому випадку величина p_i є тривалістю виконання завдання i на еталонному пристрої. Величина h_j – коефіцієнт продуктивності пристрою j (якщо $h_j > 1$, то пристрій j менш продуктивний, ніж еталонний, якщо $h_j < 1$ – більш продуктивний).

Передбачається, що всі завдання множин I_1 та I_2 надходять одночасно в нульовий момент часу, процес обслуговування кожного завдання проходить без переривань до завершення обслуговування. Пристрої можуть починати свою роботу в ненульовий момент часу.

Необхідно побудувати розклад, що мінімізує сумарне відхилення моментів закінчення виконання завдань від директивних строків.

Поставлена задача є частковим випадком задач, розглянутих у роботах [1-3]. В ній множина робіт складається з двох підмножин, кожна з яких має свій директивний строк.

2. Узагальнена схема алгоритму складання розкладу

Алгоритм розв'язання задачі складається з трьох етапів.

ЕТАП І. *Попереднє закріплення пристроїв між підмножинами завдань*

І.1 Розв'язання допоміжної оптимізаційної задачі – отримання підмножин M_1 (пристрої, на які будуть призначатися завдання множини I_1) та M_2 (пристрої, на які будуть призначатися завдання множини I_2).

ЕТАП ІІ *Побудова допустимого розкладу*

ІІ.1 Побудова допустимого розкладу S_1: за алгоритмом А1 (що описаний в розділі 5) скласти розклад виконання завдань множини I_1 пристроями множини M_1.

ІІ.2 Побудова допустимого розкладу S_2: за алгоритмом А1 скласти розклад виконання завдань множини I_2 пристроями множини M_2.

ЕТАП ІІІ. *Покращення допустимого розкладу.*

3. Задача закріплення пристроїв між підмножинами завдань

Ця задача є допоміжною, вона розв'язується на ЕТАПІ 1 алгоритму і результати її розв'язання є вхідними для наступного етапу.

Необхідно закріпити пристрої за підмножинами завдань I_1 та I_2, тобто розділити множину пристроїв M на дві підмножини M_1 та M_2, де M_1 – множина пристроїв, на які будуть призначатися завдання множини I_1, M_2 – множина пристроїв, на які будуть призначатися завдання множини I_2.

Можливі два підходи до розв'язання цієї допоміжної задачі: розподіл, який базується тільки на кількісних характеристиках множин (умовно назвемо його кількісним розподілом); розподіл, що враховує значення тривалостей та коефіцієнтів продуктивності (умовно назвемо його об'ємним розподілом).

Кількісний розподіл. Необхідно знайти такі групи пристроїв m_1 та m_2 $(m_1+m_2=m)$, щоб середня кількість завдань, що виконується різними групами пристроїв, була максимально близькою:

$$\frac{n_1}{m_1} \approx \frac{n_2}{m_2}.$$

Кількісний розподіл підходить до задачі складання розкладу для ідентичних пристроїв.

Об'ємний розподіл. Введемо додаткову характеристику пристроїв:

$s_j = \frac{1}{h_j}$, $j = 1,...,m$ – швидкість роботи пристрою j, тоді $\sum_{j\in M_1} s_j$ – сумарна швидкість пристроїв множини M_1; $\sum_{j\in M_2} s_j$ – сумарна швидкість пристроїв множини M_2.

Введемо позначення: $P_1 = \sum_{i\in I_1} p_i$, $P_2 = \sum_{i\in I_2} p_i$ – сумарні обсяги робіт з виконання завдань з множин I_1 та I_2 відповідно

Представляється логічним розбивати множину пристроїв на дві підмножини за наступним правилом: сумарна швидкість відповідної підмножини пристроїв повинна бути прямо пропорційна сумарному обсягу робіт або\та обернено пропорційна до відповідного директивного строку.

Твердження «Сумарна швидкість відповідної підмножини пристроїв є прямо пропорційною сумарному обсягу робіт» означає, що :

$$\frac{P_1}{\sum_{j\in M_1} s_j} = \frac{P_2}{\sum_{j\in M_2} s_j}. \qquad (1)$$

Твердження «Сумарна швидкість відповідної підмножини пристроїв є обернено пропорційною директивному строку відповідної множини завдань» означає, що

$$\frac{d_2}{\sum_{j\in M_1} s_j} = \frac{d_1}{\sum_{j\in M_2} s_j}. \qquad (2)$$

Твердження «Сумарна швидкість відповідної підмножини пристроїв є прямо пропорційною сумарному обсягу робіт та обернено пропорційною до відповідного директивного строку відповідної множини завдань» означає, що

$$\frac{P_1 d_2}{\sum_{j\in M_1} s_j} = \frac{P_2 d_1}{\sum_{j\in M_2} s_j}. \qquad (3)$$

З урахуванням того, що P_1, P_2, d_1 та d_2 є відомими цілими числами, а s_j, $j=\overline{1,m}$ є заданими дійсними числами, тому рівності в (1), (2) та (3) можуть бути не досяжними, отже постає задача таким чином розбити множину M на дві підмножини, щоб різниця між правими та лівими частинами цих рівнянь була мінімальною.

Таким чином, можуть бути сформульовані наступні критерії розподілу:

$$\left|\frac{\sum\limits_{j\in M_1} s_j}{\sum\limits_{j\in M_2} s_j}-\frac{P_1}{P_2}\right|\to\min$$

$$\left|\frac{\sum\limits_{j\in M_1} s_j}{\sum\limits_{j\in M_2} s_j}-\frac{d_2}{d_1}\right|\to\min$$

$$\left|\frac{\sum\limits_{j\in M_1} s_j}{\sum\limits_{j\in M_2} s_j}-\frac{P_1 d_2}{P_2 d_1}\right|\to\min$$

4. Математична модель допоміжної оптимізаційної задачі (ДОЗ)

Введемо наступні змінні:

$$x_j=\begin{cases}1, \text{ якщо пристрій } j \text{ включається до } M_1\\ 0, \text{ якщо пристрій } j \text{ включається до } M_2\end{cases},$$

$j=1,...,m$.

З урахуванням змістовного значення змінних, сумарна швидкість пристроїв множини M_1 становить $\sum\limits_{j\in M_1} s_j=\sum\limits_{j=1}^{m} s_j x_j$, множини M_2 – $\sum\limits_{j\in M_2} s_j=\sum\limits_{j=1}^{m} s_j(1-x_j)$. Тоді критерії оптимізації будуть мати вигляд:

$$\left|\frac{\sum\limits_{j=1}^{m} s_j x_j}{\sum\limits_{j=1}^{m} s_j(1-x_j)}-\frac{P_1}{P_2}\right|\to\min$$

$$\left|\frac{\sum\limits_{j=1}^{m} s_j x_j}{\sum\limits_{j=1}^{m} s_j(1-x_j)}-\frac{d_1}{d_2}\right|\to\min$$

$$\left|\frac{\sum\limits_{j=1}^{m} s_j x_j}{\sum\limits_{j=1}^{m} s_j(1-x_j)}-\frac{P_1 d_1}{P_2 d_2}\right|\to\min$$

Обмежень, окрім $x_j\in\{0;1\}$, $j=\overline{1,m}$, немає.

5. Побудова допустимого розкладу

Введемо позначення: Q_j – множина завдань, що запізнюються на пристрої j, $|Q_j|=q_j$, $j=\overline{1,m}$; $j\lfloor i\rfloor$ – номер завдання, що виконується на пристрої j та є i-тим за порядком в послідовності завдань, що запізнюються на цьому пристрої; W_j – множина завдань, що на пристрої j виконуються з випередженням, $|W_j|=w_j$, $j=\overline{1,m}$ (завдання, у якого на пристрої j випередження дорівнює 0, будемо відносити до множини W_j); $j\lceil i\rceil$ – номер завдання, що є i-тим з кінця в упорядкуванні завдань з множини W_j (на пристрої j є i-тим в послідовності завдань множини W_j, якщо рахувати від директивного строку справа наліво по часовій осі).

Визначимо сумарне відхилення моментів закінчення від директивного строку:

Сумарне випередження:

$$E=\sum_{j=1}^{m} h_j\sum_{k=1}^{w_j-1}\sum_{i=1}^{k} p_{\lceil k\rceil}=\sum_{j=1}^{m} h_j\sum_{i=1}^{w_j-1}(w_j-i)\,p_{j\lceil i\rceil}.$$

Сумарне запізнення:

$$Z=\sum_{j=1}^{m} h_j\sum_{k=1}^{q_j}\sum_{i=1}^{k} p_{j\lfloor k\rfloor}=\sum_{j=1}^{m} h_j\sum_{i=1}^{q_j}(q_j-i+1)\,p_{j\lceil i\rceil}.$$

Сумарне відхилення:

$$E+Z=\sum_{j=1}^{m} h_j\sum_{i=1}^{w_j}(w_j-i)\,p_{j\lceil i\rceil}+$$

$$+\sum_{j=1}^{m} h_j\sum_{i=1}^{q_j}(q_j-i+1)\,p_{j\lceil i\rceil}.$$

Отже, $E+Z$ являє собою суму $n=\sum\limits_{j=1}^{m}(w_j+q_j)$ доданків, кожний з яких є добутком коефіцієнту продуктивності j-го пристрою, цілого числа (що відповідає позиції завдання у розкладі відповідного

пристрою) та тривалості завдання. Назвемо *складеним коефіцієнтом* добуток коефіцієнта продуктивності пристрою на ціле число.

Для мінімізації сумарного відхилення необхідно розподіляти завдання між пристроями за правилом «найдовшому завданню повинен відповідати найменший складений коефіцієнт ЦФ» [4]. Побудований за таким правилом розклад може виявитись недопустимим (момент початку виконання розкладу на одному чи декількох пристроях може набути від'ємного значення). Тому для уникнення цієї проблеми, необхідно модифікувати алгоритм побудови розкладу.

Алгоритм А1 складання розкладу

КРОК 1. Впорядкувати завдання за незростанням тривалостей ($p_1 \geq p_2 \geq \ldots \geq p_n$).

КРОК 2. Впорядкувати значення складених коефіцієнтів ЦФ за неспаданням. Встановити їх відповідність множинам W та Q.

КРОК 3. ЦИКЛ по завданням

Для поточного завдання знайти позицію у розкладі, якій відповідає найменший вільний складений коефіцієнт

ЯКЩО складений коефіцієнт відповідає множині W та для завдання виконується умова $h_j p_{j\lceil i \rceil} \leq d - h_j \sum_{i \in W_j} p_{j\lceil i \rceil}$

ТО Завдання призначити на визначену позицію у розкладі.

ІНАКШЕ Завдання включити до множини Q на позицію з найменшим вільним складеним коефіцієнтом

6. Перестановки завдань, що покращують допустимий розклад

Для покращення розкладу пропонується здійснювати перестановки завдань між двома множинами пристроїв: M_1 – множиною пристроїв, на які (на Етапі I) призначено завдання множини I_1 та M_2 – множиною пристроїв, на які призначено завдання множини I_2.

Нехай множини I_1 та I_2 нумеровані таким чином, що $d_2 > d_1$. Пропонуються наступні перестановки.

Перестановки типу 1. Обмін завдань між пристроями множин M_1 та M_2: деяка підмножина завдань з множини I_2 може призначатися на виконання на пристрої множини M_1.

Перестановки типу 2. Обмін завдань між пристроями всередині множини M_2: деяка підмножина завдань з пристрою l може призначатися на виконання до пристрою v, $l \neq v,\ l, v \in M_2$.

Введемо наступні позначення:

$Q_j^{M_k}$ – множина завдань, що запізнюються на пристрої $j \in M_k$, $k = \{1, 2\}$, $\left|Q_j^{M_k}\right| = q_j^{M_k}$, $j = \overline{1, m}$;

$W_j^{M_k}$ – множина завдань, що на пристрої $j \in M_k$, $k = \{1, 2\}$ виконуються з випередженням, $\left|W_j^{M_k}\right| = w_j^{M_k}$, $j = \overline{1, m}$;

T_j^Q – сумарна тривалість виконання пристроєм j завдань що запізнюються;

$\Delta_j^{M_1} = d_1 + T_j^Q$ – момент закінчення виконання завдань на пристрої $j \in M_1$;

$\delta_j^{M_1} = d_2 - \Delta_j^{M_1}$ – часовий проміжок між моментом закінчення виконання завдань на пристрої j з множини M_1 та d_2.

Для спрощення запису, введемо наступне позначення коефіцієнтів цільової функції:

$z_i^{new} = \begin{cases} (w_j^{M_1} - i) \\ (q_j^{M_1} - i + 1) \end{cases}$ – коефіцієнт ЦФ для завдання, що переставляється на позицію i відносно d_2, та буде виконуватись на пристрої $j \in M_1$;

$z_i^{old} = \begin{cases} (w_j^{M_1} - i) \\ (q_j^{M_1} - i + 1) \end{cases}$ – попередній коефіцієнт ЦФ для завдання, що знаходилося на позиції i відносно d_2 та виконувалось на пристрої $j \in M_2$.

Перестановка завдання з пристрою множини M_2 на виконання до пристрою з множини M_1 відбувається, якщо виконується хоча б одна з наступних умов (рис. 1 – 3 ілюструють ці умови).

Умова 1. $p_i^{I_2} \cdot h_j^{M_1} \le \delta_j$.

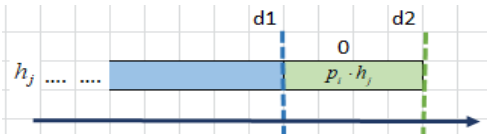


Рис. 1. Приклад перестановки згідно умови 1

Умова 2.

$\delta > 0,\ (p_i^{I_2} \cdot h_j^{M_1} - \delta_j) \cdot z_i^{new} < p_i^{I_2} \cdot h_j^{M_2} \cdot z_i^{old}$.

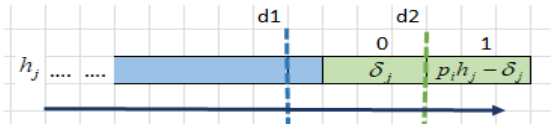


Рис. 2. Приклад перестановки згідно умови 2

Умова 3.

$\delta \ge 0, |\delta_j| + p_i^{I_2} \cdot h_j^{M_1} z_i^{new} < p_i^{I_2} \cdot h_j^{M_2} \cdot z_i^{old}$.

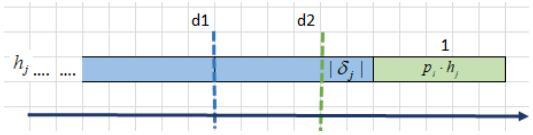


Рис. 3. Приклад перестановки згідно умови 3

Після того, як відбулися всі можливі перестановки робіт з множини I_2 на виконання до множини пристроїв M_1, перевіряємо чи можна покращити значення ЦФ шляхом перестановки робіт на множині M_2 використовуючи «місця що звільнилися».

7. Експериментальні дослідження

Для перевірки ефективності розроблених критеріїв оптимізації допоміжної задачі була проведена серія експериментів. Для виявлення впливу особливостей задач складання розкладів були виділені наступні ознаки, за якими задачі можна класифікувати:

- *задачі класу 1*: $P_1 < P_2$, $d_1 < d_2$;
- *задачі класу 2*: $P_1 \approx P_2$ (сумарні тривалості робіт з множин I_1 та I_2 є приблизно однаковими), $d_1 < d_2$.

В таблиці 1 представлено параметри генерації індивідуальних задач. Усі величини, що задані у проміжку, розподілені рівномірно. При дослідженні було згенеровано по 500 індивідуальних задач з кожного класу.

На рисунках 4 та 5 представлені результати розрахунків: значення цільової функції (ЦФ) в початкових розкладах згенерованих індивідуальних задач в залежності від критерію оптимізації ДОЗ.

Узагальнення результатів експериментів дало наступне: для задач класу 1 відносна кількість випадків (у %) побудови початкового розкладу з кращим значенням ЦФ залежно від використаного в ДОЗ критерію закріплення пристроїв за множинами завдань становила 42% для критерію 1, 20% для критерію 2, 38% для критерію 3. Для задач класу 2: 36% для критерію 1, 10% для критерію 2, 54% для критерію 3.

Згідно отриманих результатів можна зробити наступні висновки: для індивідуальних задач класу 1 при розв'язанні ДОЗ доцільно використовувати 1-й критерій оптимізації, а для задач класу 2 – 3-й критерій.

Табл. 1. Параметри генерації індивідуальних задач

	m	*h*	d_1	d_2	n_1	n_2	p_i ($i \in I_1$)	p_i ($i \in I_2$)
Клас 1	10	1-4	600	700	100	150	1-80	1-80
Клас 2	10	1-4	600	700	100	150	1-40	40-80

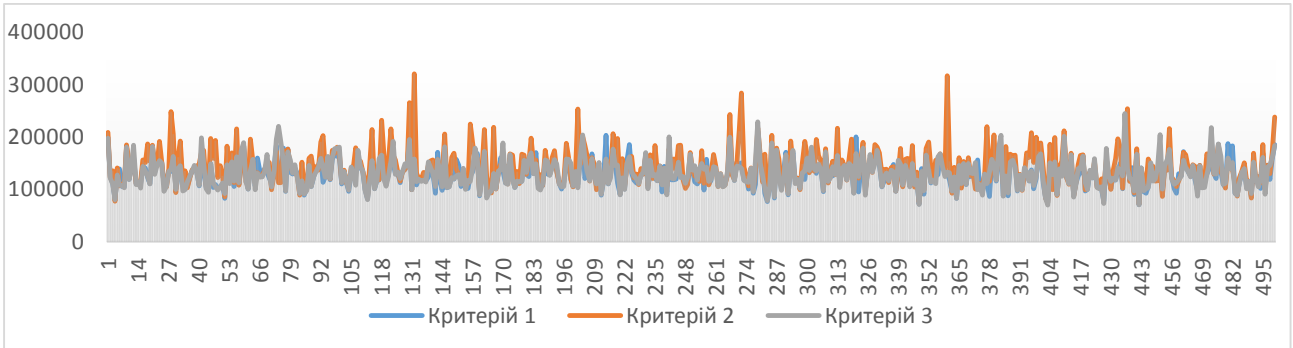


Рис. 4. Залежність значення ЦФ початкового розкладу в залежності від критерію ДОЗ для задач класу 1

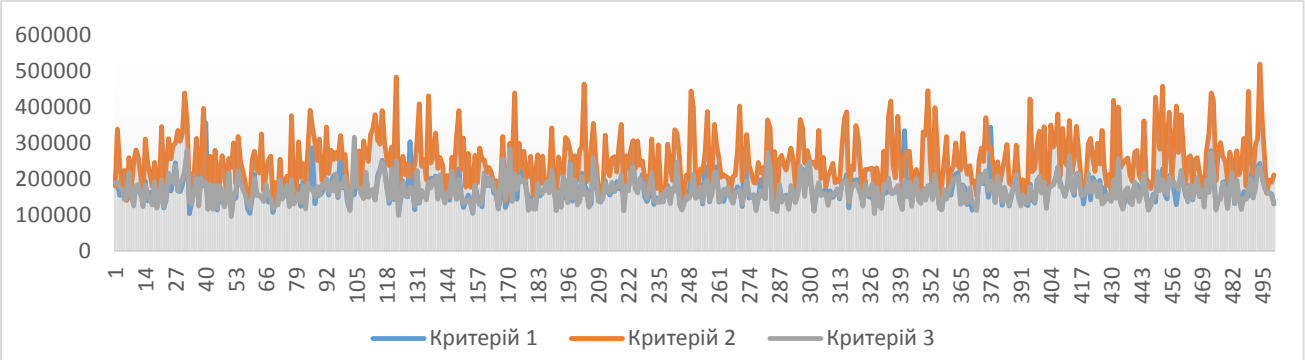


Рис. 5. Залежність значення ЦФ початкового розкладу в залежності від критерію ДОЗ для задач класу 2

Перелік посилань

1. Mensendiek, A., Gupta, J., & Herrmann, J. (2015). Scheduling identical parallel machines with fixed delivery dates to minimize total tardiness. European Journal of Operational Research, 243(2), 514-522.
2. Kayvanfar, V., Komaki, G., Aalaei, A., & Zandieh, M. (2014). Minimizing total tardiness and earliness on unrelated parallel machines with controllable processing times. Computers & Operations Research, 41, 31-43.
3. Sivrikaya-Şerifoğlu, F., Ulusoy, G. (1999). Parallel machine scheduling with earliness and tardiness penalties. Computers & Operations Research, 26(8), 773-787.
4. Задача мінімізації сумарного відхилення від спільного директивного строку при виконанні завдань паралельними пристроями / А. В. Годна, А. О.Маленко, О. Г. Жданова, М. О. Сперкач. // Науковий огляд. – 2017. – №9(14). – С. 14–32.

УДК 683.519

ГРАЧОВА О.А.
ФІНОГЕНОВ О.Д.

МЕТОД АНАЛІЗУ ІЄРАРХІЙ ПРИ АНАЛІЗІ ДАНИХ, ЗАЛЕЖНИХ У ЧАСІ

Темою даної статті є практичне використання методу аналізу ієрархій для аналізу даних, що є залежними у часі. В якості прикладу обрано часовий ряд погоди міста Києва. Демонструється побудова ієрархічної моделі, оцінка критеріїв, розрахунок абсолютних відхилень альтернатив за критеріями та вибір найкращої альтернативи за результатами проведеного аналізу.

The subject of the article is application of the analytic hierarchy process to an analysis of the data which has time relations. As an example, the time series of weather data in the city of Kyiv is demonstrated. The article shows hierarchy modelling, ranking of criteria, calculation of absolute deviation and decision making based on the results of analysis.

1. Вступ

У сучасному світі зі збільшенням негативних факторів впливу все важливіше місце займає дослідження здоров'я людини. Першим кроком до усунення негативних ефектів є відповідні дії людини на погодні умови, прогнозування яких є вкрай трудомісткою та в багатьох випадках нечіткою задачею. На даний момент, все більше людей отримують дані погоди завдяки мережі інтернет, але на багатьох сайтах, що інформують про погодні умови, є відмінності як у показниках, так і в наведених даних. Вибір одного з декількох джерел інформації про погоду можна представити у вигляді багатокритеріальної задачі, одним з методів вирішення якої є метод аналізу ієрархій (MAI), що містить аналіз не лише кількісних, але і якісних показників. Вибір людиною погодних сайтів також обумовлений великою інерцією – звичне налаштування або інтуїтивно зрозуміле подання інформації може бути більш впливовим фактором, а ніж достовірність.

2. Термінологія

Метод аналізу ієрархій — замкнута логічна конструкція, яка виконує аналіз складних проблем за допомогою простих правил. Метод заснований на парних порівняннях альтернативних варіантів за різними критеріями з використанням бальної шкали (як правило, від 1 до 9 балів) і надалі ранжируванні набору альтернатив за всіма критеріями і цілях. Зв'язки між критеріями враховуються шляхом побудови ієрархії критеріїв і застосування парних порівнянь для визначення важливості критеріїв і підкритеріїв.

3. Опис ходу досліджень

У задачі, що є предметом досліджень, вихідними множинами елементів було обрано дані про температуру (T), вологість (H), швидкість і спрямованості вітру(W), атмосферний тиск (P), імовірність опадів (Pr), хмарність (C) та оціночну температури (FL) за чотири періоди доби: ранок, день, вечір та ніч. Оскільки дані на сайтах можуть оновлюватися у режимі реального часу, показники знімаються двічі на добу - вранці і ввечері, що дозволяє в подальшому аналізувати достовірність (рис.1). Для спрощення аналізу будемо вважати, що геометрично населений пункт являє собою точку, оскільки через великі розміри міста показники у різних районах можуть відрізнятися.

Інформація, що подається на сайтах, порівнюється із даними європейської метеостанції [3] у вигляді ступеню абсолютного відхилення за допомогою парних порівнянь.

Рис. 1. Приклади стислих прогнозів погоди

Угруповання даних протягом доби доцільно проводити тетрадами: по чотири показники кожного періоду доби об'єднуються в групу - це формує другий рівень ієрархії. Оскільки ознаки впливають на досягнення кінцевої мети нерівномірно, необхідно визначити інтенсивність впливу ознак на кінцеву оцінку альтернативи.

У цьому завданню метою є вибір найкращої з п'яти альтернатив, якими є п'ять популярних погодних сайтів: Gismeteo[1], Meta[2], Meteo[4], Rp5[5] та Sinoptik[6] за період з 1 по 7 січня 2018 року (рис.2, 3).

З огляду на вищеописаний принцип угруповання, першим рівнем ієрархії буде набір з 28 компонент. Другий рівень ієрархії буде являти собою угруповання критеріїв за ознакою схожості.

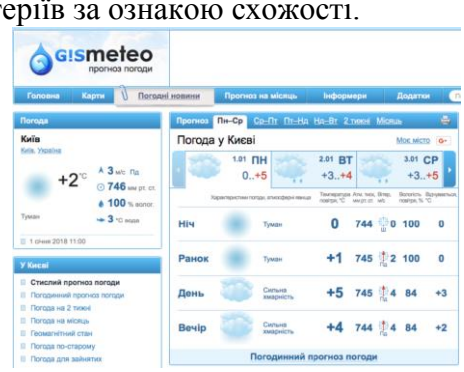


Рис. 2. Приклад розгорнутих даних погоди

Таким чином, другий рівень ієрархії являє собою 7 компонент. Описана ієрархічна структура проілюстрована на рис.1, де маркування T1-4 відповідає показникам температури, *P1-4* — атмосферного тиску, *H1-4* — вологості, *W1-4* — вітру, *FL1-4* — відчувається як, *Pr1-4* — імовірності опадів, *C1-4* — хмарності.

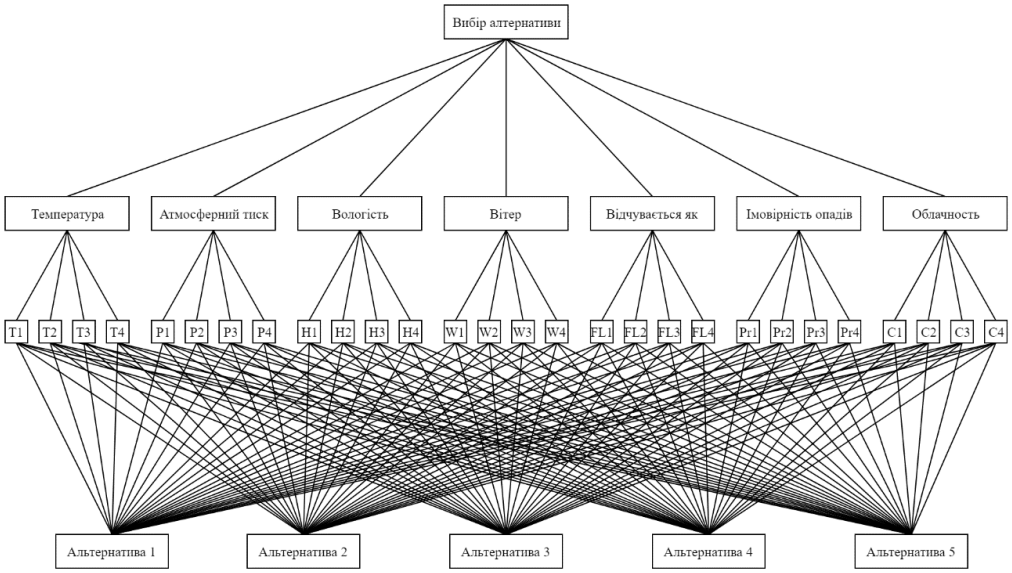


Рис. 3. Ієрархічна структура

Отже, постановка задачі багатокритеріального вибору може бути сформульована наступним чином - необхідно знайти аналітичну оцінку $y^{(m)}$:

$$y^{(j-1)} = \{y_i^{(j-1)}\}_{i=1}^{n^{(j-1)}}, j \in [2, m] \quad (1.1)$$

і якісну оцінку ефективності кожного альтернативного рішення у вигляді загального критерію прийняття рішення:

$$p^{(j-1)} = \{p_k^{(j-1)}\}_{k=1}^{n^{(j-1)}}, j \in [2, m] \quad (1.2)$$

і вибрати найкращу[7, 8].

Оскільки деякі з альтернатив не надають повний набір показників для аналізу, ієрархія є неповнозв'язанною. Для формування повнозв'язанної ієрархії загального виду доцільно ввести оцінку відхилення даних у шкалі оцінок 1÷5 або менше, а для даних, які відсутні для альтернативи, використовувати оцінки 1/8,1/9, щоб мінімізувати вплив на загальну оцінку [5, 6]. Отримана матриця критеріїв наведена у таблиці 1.

Табл. 1. Оцінка критеріїв

Критерії	**Температура**	**Атмосферний тиск**	**Вологість**	**Вітер**	**Імовірність опадів**	**Хмарність**	**Відчувається як**
Температура	1	1/4	2	1/4	8	1/5	9
Тиск	4	1	1/3	1/5	8	1/5	9
Вологість	1/2	3	1	1/6	8	1/3	9
Вітер	4	5	6	1	8	1/6	9
Імовірність опадів	1/8	1/8	1/8	1/8	1	7	9
Хмарність	5	5	3	6	1/7	1	9
Відчувається як	1/9	1/9	1/9	1/9	1/9	1/9	1

Після аналізу отриманих даних, було розраховано абсолютні відхилення для кожного з ресурсів за п'ятьма показниками (таблиця 2). Для імовірності опадів та критерію «відчувається як» розрахунки не проводилися, адже міжнародна станція не надає цієї інформації.

Табл. 2. Результати розрахунку абсолютних відхилень

Альтернатива	**Температура**	**Хмарність**	**Атмосферний тиск**	**Вологість**	**Вітер**
Gismeteo	0.214	0.025	33.785	0.003	0.75
Meta	0.0	0.042	23.857	0.014	0.5
Meteo	0.785	0.022	40.571	0.022	0.536
Rp5	0.857	0.003	26.285	0.087	0.5
Sinoptik	0.785	0.012	33.214	0.009	0.321

За результатами показників абсолютного відхилення показників температури найкращою альтернативою виявився сайт Meta, за показниками вологістю – Gismeteo, за показниками атмосферного тиску - Meta, за показниками хмарності - Rp5 та за вітром – Meta.

Після розрахування власного вектору матриці, було отримано результат, наведений у таблиці 3.

Легко помітити, що за результатами приорітизації найбільш важливими показниками є вітер, хмарність та тиск. В двох з трьох основних критеріях Meta надає найбільш достовірні дані, що дозволяє зробити висновок, що Meta є найкращою альтернативою.

Таблиця 3. Результати розрахунку власного вектору матриці критерії

Критерій	**Приорітет**
Температура	0.141
Тиск	0.155
Вологість	0.149
Вітер	0.226
Імовірність опадів	0.119
Хмарність	0.198
Відчувається як	0.011

4. Висновки

Результатом аналізу MAI є глобальні ваги альтернатив, що отримані за результатами аналізу даних за добу. Вибір сайту з множини ранжувань за деякий часовий проміжок в першому наближені можна проводити за модою ряду найкращих альтернатив. На проміжку у 7 днів найкращою альтернативою виявився ресурс Meta.

Література

1. **Gismeteo** [Електронний ресурс] // Режим доступу**:** https://www.gismeteo.ua/ua/
2. Meta Погода [Електронний ресурс] // Режим доступу: http://pogoda.meta.ua/
3. World Weather Online [Електронний ресурс] // Режим доступу: https://www.worldweatheronline.com/
4. Meteo [Електронний ресурс] // Режим доступу: https://www.meteo.ua/
5. Rp5 [Електронний ресурс] // Режим доступу: http://rp5.ua/
6. Sinoptik [Електронний ресурс] // Режим доступу: https://ua.sinoptik.ua/
7. Саати Т. Аналитическое планирование / Т. Саати, К. Кернс. – М.: Радио и связь, 1991. – 320 с. – С. 96.
8. Саати Т. Принятие решений. Метод анализа иерархий / Т. Саати. – М.: Радио и связь, 1993. – 210 с.

УДК 004.94:519.876.5

ДИФУЧИН А. Ю.,
ТОМАШЕВСЬКИЙ В. М.

ВЕБ-СЕРВІС МОДЕЛЮВАННЯ ДИСКРЕТНО-ПОДІЙНИХ СИСТЕМ

У даній статті розкриті основні проблеми реалізації веб сервісу моделювання дискретно-подійних систем. Розглянуто два підходи до моделювання: за допомогою стохастичних мереж Петрі та з використанням Петрі-об'єктної технології. Результати порівняння цих підходів показали, що алгоритм імітації Петрі-об'єктної моделі має меншу обчислювальну складність та за часом моделювання приблизно в два рази швидше за алгоритм імітації стохастичних мереж Петрі. Також важливу роль у часі моделювання відіграє мова реалізації алгоритму імітації. Показано, що реалізація Петрі-об'єктної технології моделювання на мові Java працює в 70 разів швидше за ідентичну реалізацію на мові Ruby. Результатом даної роботи є веб-сервіс моделювання дискретно-подійних систем з використанням Петрі-об'єктної технології.

This article shows the main design problems of the web service for discrete event systems simulation. Two simulation approaches are considered: stochastic Petri Net approach and Petri-object model approach. The result of comparing these approaches shows that Petri-object model simulation algorithm has lower computational complexity than stochastic Petri Net and is approximately two times faster. The very important part is the choice of the programming language used for the realization of simulation algorithm. The realization made with Java is 70 times faster than the one made with Ruby. Web service for discrete event system simulation with use of Petri-object model approach was developed as a result of this work.

1. Вступ

Моделювання дискретно-подійних систем є основою для систем прийняття рішень та управління інформацією. Від якості побудованої моделі значною мірою залежить якість керуючих процесів. Отримання результатів процесу моделювання є основним завданням моделювання, тому розробка програмного забезпечення з імітаційного моделювання є дуже важливим фактором. Високий рівень технічної складності систем призводить до використання підходів які мають певні характеристики. Системи моделювання вимагають побудову моделі з однаковими елементами, гнучке моделювання динамічних елементів, візуальне зображення моделі і можливості його коригування та модифікації. Гнучкість моделювання передбачає, що дослідники деталізують процес до найменшого елемента, але вимоги до зручності представлення моделі спричиняють їх більш абстрактне визначення.

Залежно від того, яким чином відтворюються в часі стани моделі, розрізняють дискретні, неперервні й дискретно-неперервні (комбіновані) моделі [1]. Очевидно, що мова йде про динамічні моделі, оскільки оперуємо поняттям часу. Дискретна система – це така система, в якій змінні стану змінюються лише в дискретні моменти часу. Неперервна система – це система, в якій змінні стану змінюються безперервно з плином часу. Прикладами дискретної системи можуть слугувати релейно-контактні схеми, цифрові системи комутації; неперервної – механічні та термодинамічні фізичні системи. В даній роботі розглядатимемо тільки дискретні системи. Дискретні динамічні системи, в свою чергу, можна поділити на подійні (дискретно-подійні), в яких зміна станів повністю залежить від настання дискретних подій, та часові, в яких зміна станів відбувається через фіксовані інтервали часу (такти). Надалі будемо розглядати лише дискретно-подійні системи.

2. Моделювання дискретно-подійних систем

Теоретичні основи та специфікація дискретно-подійних систем, що описані в [1], є узагальненням теорії систем масового обслуговування і використовуються більшістю програмного забезпечення для моделювання дискретно-подійних систем, включаючи такі відомі комерційні проекти як Arena, ExtendSim, Plant Simulation та ін. Система являє собою набір блоків, що реалізують типові функції такі, як очікування, обробка, транспортування та інші. Однак в цьому випадку програмування керуючих елементів може викликати певні труднощі, оскільки алгоритми, що реалізують ці блоки не можуть бути модифіковані. Тому такі задачі як моделювання кібератак, моделювання дорожньо-транспортного руху не можуть бути представлені блоками моделі роботи підприємства.

Альтернативний підхід передбачає використання мереж Петрі. Такий підхід має перевагу перед іншими системами моделювання, оскільки в основі нього лежить апарат математичного моделювання. Мережі Петрі являють собою орієнтований дводольний граф з переходами стану. Переходи представляють події системи, а позиції являють собою умови, які призводять до настання події. Направлені дуги з'єднують переходи з позиціями, які мають маркери і навпаки. Перехід відбувається, коли для кожної позиції виконується умова: кількість маркерів щонайменше дорівнює вазі дуги, яка веде від вхідної позиції до переходу. Виконання переходу виконується шляхом видалення маркерів у вхідних позиціях та додавання маркерів у вихідних позиціях відповідно до ваги дуги.

Вихід маркерів відбувається з визначеною тимчасовою затримкою для мережі Петрі. Часова затримка може бути задана стохастичною величиною.

Позиція може бути з'єднана з переходом за допомогою інформаційної дуги. Це означає, що вихід маркерів з такої позиції не виконується при виконанні переходу. Наприклад, коли автомобіль рухається по перехрестю стан "зеленого світла" все ще зберігається. Інший приклад, коли зловмисник використовує вразливість, це не означає, що вразливість зникла, тому відповідний стан не повинен змінюватись.

3. Огляд аналогів

На сьогодні існує багато Петрі-імітаторів, перелік яких представлений в [2]. В таблиці 1 наведено порівняння характеристик, недоліки та переваги розробленого веб-сервісу моделювання дискретно-подійних систем з використанням мереж Петрі з найбільш розповсюдженими на даний момент Петрі імітаторами.

Табл. 1. Порівняння Петрі-імітаторів [3]

	Об'єктно-орієнтовний підхід	**Стохастичні мережі Петрі**	**Веб-орієнтований графічний редактор**	**Швидке моделювання**	**Анімація**
Coopn	+	-	-	+	+
JSARP	+	-	-	+	+
PNTalk	+	-	-	+	+
Renew	+	-	-	+	+
CPN Tools	-	+	-	+	+
Petri.NET Simulator	-	-	-	+	+
WoPeD	-	-	-	+	-
PIPE2	-	+	-	+	+
DESS	+	+	+	+	+

Найбільшою перевагою DESS є унікальний об'єктно-орієнтовний підхід до побудови моделі з використанням мереж Петрі [4]. Ця концепція дозволяє представляти мережу Петрі як параметр конструктора об'єкта. Це дозволяє будувати багато об'єктів на основі вже побудованої мережі Петрі. Користувач може швидко створити список Петрі-об'єктів, а потім встановити зв'язок між ними. Також розроблена інша унікальна особливість - веб-графічний редактор мереж Петрі, за допомогою якого можна створювати або редагувати мережі Петрі на будь-якому комп'ютері або навіть на смартфоні чи планшеті, які підключені до мережі Інтернет. Єдиною вимогою є наявність сучасного браузера з підтримкою Javascript та стандарту HTML 5. Дивлячись на найбільш популярні об'єктно-орієнтовані Петрі

імітатори можна побачити, що стохастичні мережі Петрі підтримують лише деякі з них. Підтримка стохастичних мереж Петрі є дуже важливою для реалістичного моделювання складних систем. Отже, можна побачити, що DESS має ряд важливих переваг перед іншими Петрі імітаторами, які полегшують процес моделювання систем.

4. Веб-сервіс моделювання дискретно-подійних систем

Програмне забезпечення для моделювання дискретно-подійних систем розроблено на базі клієнт-серверної архітектури.

Клієнтська частина розроблена у вигляді браузерного застосунку, написаного на мові Javascript з використанням бібліотеки ReactJS, та підтримує роботу інтерфейсу користувача. Графічний інтерфейс забезпечує побудову Петрі-об’єктної моделі з використанням графічних елементів стохастичних мереж Петрі: позиції, переходи, дуги. Будь-який перехід повинен бути визначений наступним набором параметрів: значення часу затримки, значення пріоритету та ймовірності. Значення затримки часу може бути задано стохастичною або детермінованою невід'ємною величиною. Значення пріоритету задається позитивним цілим числом. Значення ймовірності має знаходитись в інтервалі [0; 1]. За замовчуванням параметри переходу мають нульову затримку часу, пріоритет 1 та ймовірність 1.0. Кількість маркерів повинна бути визначеною для кожної позиції. Значення за замовчуванням для цього параметра дорівнює нулю. Будь-яка дуга визначається кількістю посилань і булевим значенням, що вказує, чи є дуга інформацією. За замовчуванням дуга є неінформаційною з одним зв’язком.

Всі графічні елементи підтримують базові маніпуляції: створення, перетягування, видалення та редагування параметрів.

Серверна частина складається з Ruby on Rails застосунку та Java мікросервісу. Ruby on Rails використовується для обробки клієнтських запитів, роботою з СУБД, управлінням облікових записів користувачів, оскільки цей фреймворк надає зручні інструменти саме для цього набору задач.

Для реалізації алгоритму імітації вирішено створити окремий мікросервіс, написаний на мові Java, оскільки мова Java вирізняється високою продуктивністю, яка необхідна для роботи складних алгоритмів бібліотеки PetriObjLib. Також використання мови Java дозволяє розпаралелити алгоритм імітації в майбутньому. Комунікація між клієтським ReactJS застосунком, Ruby on Rails застосунком та Java мікросервісом відбувається через протокол HTTP з даними у форматі JSON.

Динаміку Петрі-об’єктної моделі можна досліджувати, виконуючи її запуск. При запуску модель з клієнтського застосунку передається по HTTP протоколу на сервер у форматі JSON. Ruby on Rails REST API зберігає або оновлює модель в обліковому записі користувача і передає її далі на обробку в Java мікросервіс, який запускає безпосередньо процес імітації. Після виконання імітації результати повертаються на Ruby on Rails API. Результати моделювання включають протокол події, середнє, максимальне і мінімальне значення в позиціях та середнє, максимальне і мінімальне навантаження на переходи. Панель звітів дозволяє користувачеві переглянути інформацію про всі події, що сталися під час моделювання та статистику для кожного елемента мереж Петрі.

Петрі-об'єкти створюються та зберігаються в списку об’єктів моделі. Зв'язки між Петрі-об'єктами визначаються візуально. Користувачі можуть обрати Петрі-об'єкт зі списку і визначити зв'язок з іншими Петрі-об'єктами.

Авторизація користувача виконується через внутрішній обліковий запис. Авторизований користувач має право створювати та зберігати моделі у своєму обліковому записі.

Веб-реалізація забезпечує кросплатформну роботу і може підтримувати розширені можливості для співпраці та колективної розробки моделей.

Експериментальне дослідження часових показників було виконано з Петрі-об'єктною моделлю, що складалась з n Петрі-об'єктів з k = 5 послідовних подій для кожного. Загальні позиції використовуються для послідовного з'єднання об'єктів. Така конструкція моделі дозволяє гнучко

контролювати кількість події nk. Беручи один Петрі-об'єкт nk з послідовними явищами отримаємо стохастичну мережу Петрі. Це дозволяє порівнювати реалізації Петрі-об'єктної моделі і стохастичної мережі Петрі. Результати показані на рис. 7 підтверджують теоретичну поліноміальну оцінку моделі складність якої дана в [5]. Алгоритм моделювання Петрі-об'єктної моделі забезпечує скорочення часу виконання в два рази в порівнянні зі стохастичними мережами Петрі.

Рис. 1. Залежність часу моделювання від складності моделі

Табл. 2. Порівняння часу виконання алгоритму імітації на мові Ruby та Java [3]

Параметри		tRuby			tJava			tRuby/tJava	
Кількість об’єктів	Кількість переходів, n	Час викон. Петрі-об’єктної моделі, tPO	Час викон. Мережі Петрі, t	t/tPO	Час викон. Петрі-об’єктної моделі, tPO	Час викон. Мережі Петрі, t	t/tPO	Час викон. Петрі-об’єктної моделі, tPO	Час виконання. Мережі Петрі, t
20	100	343.73	548.95	1.6	4.54	8.37	1.8	76	66
40	200	1238.15	2094.15	1.7	17.11	32.22	1.9	72	65
60	300	2862.23	6080.37	2.1	38.88	69.61	1.8	74	87
80	400	4920.18	11206.46	2.3	70.90	118.73	1.7	69	94
100	500	6917.14	14131.52	2.0	122.64	177.73	1.4	56	80

Таблиця 2 показує порівняння реалізацій алгоритму імітації Петрі-об’єктної моделі на мовах Java та Ruby. Коефіцієнт зменшується приблизно в два рази при використанні алгоритму імітації Петрі-об'єктної моделі.

Однак реалізація Ruby в 70 разів повільніше в порівнянні з реалізацією на Java. Такий результат обумовлений тим, що Ruby є інтерпретованою мовою на відміну від Java.

5. Висновки

Петрі-об’єктна реалізація дискретно-подійних систем дозволяє описувати динаміку системи за допомогою єдиного представлення стохастичних мереж Петрі.

Мережа Петрі створюється з Петрі-об'єкта, який містить багато елементів з подібною динамікою, використовуючи об'єктно-орієнтований підхід. Шляхом розподілу моделі на Петрі-об’єкти досягається значне зменшення обчислюваної складності алгоритму імітації.

Технологія побудови Петрі-об'єкта дозволяє користувачеві сконцентруватись, по-перше, на описі базової динаміки елементів моделі, по-друге, на створенні елементів з заданими параметрами, на описі динаміки самої моделі.

Розроблений веб-сервіс не тільки спрощує процес побудови моделі, але також забезпечує незалежність від ресурсів клієнтської машини, оскільки всі обрахунки є хмарними.

Список літератури

1. Томашевський В. М. Моделювання систем / В. М. Томашевський. – Київ : Видавнича група BHV, 2005. – 352 с.
2. Petri Nets Tools Database Quick Overview, [Електронний ресурс] //Режим доступу: https://www.informatik.unihamburg.de/TGI/PetriNets/tools/quick.html / accessed 11/03/2017
3. Stetsenko Inna V., Dyfuchyn, A., Leshchenko, K., John, D., Web application for visual modeling of discrete event systems // Internet Technologies and Applications, ITA 2017 - Proceedings of the 7th International Conference 8101916, с. 86-91
4. I.V. Stetsenko, “Theoretical basis of Petri-object simulation,” Mathematical Machines and Systems, no. 4, 2011, pp.135-150 (in Russian)
5. I.V. Stetsenko, V. Dorosh, A. Dyfuchyn “Petri-Object Simulation: Software Package and Complexity,” Proceedings of the 8th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS’2015), Warsaw (Poland), 2015, pp. 381-385.

УДК 004.031.42

КОКШАЙКИНА М.М.
ГОЛОВЧЕНКО М.М.

МОБІЛЬНИЙ ЗАСТОСУНОК ДЛЯ ПОШУКУ ЗОБРАЖЕНЬ НА ТЕЛЕФОНІ ПО ТЕКСТОВОМУ ВМІСТУ

В даній статті описана ідея мобільного застосунку для роботи з зображеннями на телефоні, основні вимоги, модульна структура та технології, обрані для розробки.

In this article described the idea of the mobile application for working with images on a mobile device, main requirements, module structure and technologies chosen for development.

Вступ

У наш час швидкого розвитку інформаційно-комунікаційних технологій можливість швидко знаходити та обмінюватися інформацією необхідна як ніколи до цього. Сучасні мобільні пристрої нині виконують не тільки функцію комунікації, сучасний мобільний телефон є потужним обчислювальним пристроєм з гнучкою операційною системою та великою пам'яттю, що дозволяє використовувати його для обробки та зберігання різних даних.

Одним з типів інформації, що постійно створюється, використовується і передається користувачами, є зображення. Сучасні камери мобільних телефонів мають технічні можливості створити якісні фото, що полегшує процес збереження та використання інформації. Проте звичка користувачів постійно зберігати необхідні дані в вигляді фото часто призводить до того, що знайти необхідну інформацію серед великої кількості зображень на телефоні стає складно.

Метою розробки є створення мобільного застосунку, що полегшує пошук необхідних зображень на телефоні (або іншому пристрої), а з тим і доступ до потрібної інформації шляхом розпізнавання текстового вмісту на зображенні, збереженні отриманого опису зображень, і подальшого пошуку зображень вже по текстовому опису.

1. Огляд існуючих технологій та програмних продуктів

Технологія OCR (Optical Character Recognition) - широко відома і використовувана технологія механічного чи електронного перетворення зображень друкованого, рукописного або друкованого тексту в машино-кодований текст. Це поширений метод оцифровки друкованих текстів, для подальшого редагування в електронному вигляді, пошуку, компактнішого зберігання, відображення. Технологія OCR є предметом досліджень у галузі розпізнавання образів, штучного інтелекту та комп'ютерного бачення.

Ранні версії систем, що використовують технологію, потребували навчання на зображеннях кожного шрифту, і вони працювали з одним шрифтом за раз. Розширені системи, здатні показувати високу точність розпізнавання для більшості шрифтів, тепер є загальними та підтримуються різними цифровими вхідними файлами формату файлів. Більшість систем розпізнають друкований текст, проте деякі можуть працювати і з рукописним.

Важливо пам'ятати, що будь-яка система оптичного розпізнавання не ідеальна, і можливо, що текст буде розпізнаний не точно. Тим не менш, OCR системи постійно вдосконалюються та налаштовуються для кращої точності.

Існує декілька відомих технічних рішень у цій предметній області. Всі вони використовують технологію OCR (оптичний читач символів). Інтегрована з мобільними додатками, вона дозволяє конвертувати відскановане зображення в текст миттєво без необхідності вводити текст. Далі розглянуті 3 основні з них.

1. Tesseract - відома бібліотека OCR з відкритим кодом, яку можна інтегрувати з іншими программами і додатками. Спочатку була розроблений Hewlett Packard Labs, а потім була випущена як

вільне програмне забезпечення відповідно до ліцензії Apache 2.0 у 2005 році. Розробка була спонсорована компанією Google з 2006 року. Дозволяє використовувати OCR для побудови багатьох програм і додатків, наприклад, сканування візитних карток та імпорту тексту до списку контактів, сканування зображення та відтворення тексту для людей з вадами зору тощо. Підтримує 101 мову (друковану), серед яких англійська, російська та українська. Може бути використано офлайн. Безкоштовно для використання.

2. Google Mobile Vision API - фреймворк для пошуку об'єктів у фотографіях та відео. В основу входять детектори, які визначають і описують візуальні об'єкти у зображеннях або відеокадрах, а також події, керовані API, який відстежує положення цих об'єктів у відео. В даний час Mobile Vision дозволяє працювати з обличчями, штрих-кодами та текстом на зображеннях. Підтримує 19 мов (друкованих), серед яких англійська. Може бути використано офлайн. Безкоштовно для використання.

3. Google Cloud Vision API. Дозволяє зрозуміти зміст зображення, інкапсулюючи потужні моделі машинного навчання в простий у використанні REST API. Швидко класифікує зображення у тисячі категорій (наприклад, "вітрильник", "лев", "Ейфелева вежа"), виявляє окремі об'єкти та обличчя в зображенні та знаходить та читає друковані слова, що містяться в зображенні. Дозволяє виявляти текст на зображеннях разом із автоматичною ідентифікацією мови. API Vision підтримує широкий набір мов. Тільки онлайн використання. Безкоштовне розпізнавання тільки до 1000 зображень на місяць.

Також існує досить багато мобільних застосунків (наприклад Evernote, Google Keep, Google Goggles, OneNote), які певним чином працюють з розпізнаванням тексту на зображеннях. Але усі вони вимагають створення окремого “документу” всередині застосунку і не працюють напряму з усіма фото у галереї (пам’яті). Тому створення застосунку який працював би відразу з усіма зображеннями у галереї було б актуальним.

2. Вимоги до розроблюваного застосунку

Для розробки застосунку було вирішено обрати платформу Android - найпопулярнішу мобільну операційну систему. Застосунок має розпізнавати друкований текст англійською мовою. Для розпізнавання обрано використати Google Mobile Vision API фреймфорк.

До застосунку були висунуті такі функціональні вимоги:

Табл 1. Функціональні вимоги

Варіант використання	Функціональна вимога
Попередня робота з зображеннями	Розпізнавання англійського друкованого тексту на зображенні
	Редагування розпізнаного тексту
	Збереження розпізнаного тексту
	Видалення тексту зображень, що більше не існують
Пошук	Пошук зображень по текстовому вмісту

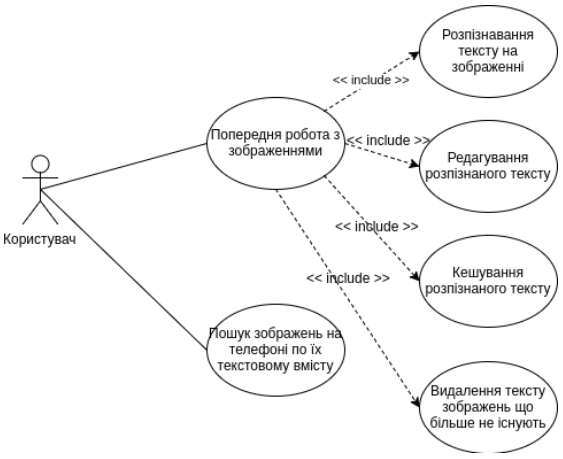


Рис. 1. Діаграма варіантів використання

Також були виділені наступні нефункціональні вимоги:

1. Застосунок повинен працювати на пристроях з операційною системою Android 5 та вище.
2. Збереження результатів розпізнавання тексту має проводитись у вбудованій базі даних SQLite.
3. Застосунок повинен мати простий та інтуїтивно зрозумілий інтерфейс.

3. Постановка комплексу завдань розробки

Для досягнення поставленої мети повинні бути вирішені наступні задачі:

1. Створення модулю розпізнавання тексту на зображеннях
2. Створення зручного інтерфейсу користувача для задоволення функціональних вимог

Застосунок має складатися з 2 основних модулів: інтерфейсу користувача та модулю розпізнавання тексту на зображеннях.

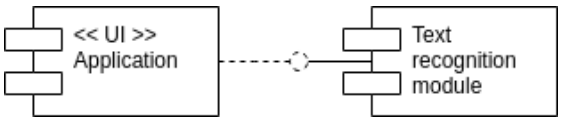


Рис. 2. Діаграма компонент

Інтерфейс користувача повинен базуватися на стандартних елементах Android застосунків для забезпечення коректної роботи на будь-якій версії операційної системи та бути зручним та інтуїтивно зрозумілим.

Модуль розпізнавання тексту на зображеннях для роботи повинен використовувати OCR технологію та Google Mobile Vision API.

Висновок

Підсумовуючи попередні розділи, у статті було детально розглянуто і проаналізовано предметне середовище - ринок мобільних застосунків та технологію OCR, оглянуті існуючі технічні рішення (Tesseract, Google Mobile Vision API, Cloud Vision API) та відомі програмні продукти. Виявлено що усі програмні продукти вимагають створення окремого “документу” всередині застосунку і не працюють напряму з усіма фото у галереї (пам’яті). Тому було вирішено що окремий застосунок, що працюватиме відразу з усіма зображеннями у пам’яті пристрою буде актуальним. Далі було досліджено сценарій користувача, розроблені функціональні та нефункціональні вимоги. Вибрано платформу для розробки (Android) та технологію розпізнавання тексту на зображеннях (Google Mobile Vision API). Сформульовані задачі розробки.

Список літератури

1. Optical Character Recognition. [Електронний ресурс] // Режим доступу: https://en.wikipedia.org/wiki/Optical_character_recognition.
2. Tesseract. [Електронний ресурс] // Режим доступу: https://github.com/tesseract-ocr/tesseract.
3. Google Mobile Vision API. [Електронний ресурс] // Режим доступу: https://developers.google.com/vision/
4. Google Cloud Vision API. [Електронний ресурс] // Режим доступу: https://cloud.google.com/vision/

УДК 519.854.2

МАЛЕНКО А.О.
ЖДАНОВА О.Г.
СПЕРКАЧ М.О.

ЗАДАЧА МІНІМІЗАЦІЇ СУМАРНОГО ВІДХИЛЕННЯ ВІД СПІЛЬНОГО ДИРЕКТИВНОГО СТРОКУ ПРИ ВИКОНАННІ ЗАВДАНЬ ПАРАЛЕЛЬНИМИ ПРИСТРОЯМИ

У роботі розглянуто задачу теорії розкладів, у якій необхідно скласти розклад виконання завдань зі спільним директивним строком паралельними пристроями за критерієм мінімізації сумарного відхилення моментів завершення завдань від директивного строку. З урахуванням умов поліноміальної та неполіноміальної розв'язуваності задач розроблено алгоритми складання розкладу для пропорційних пристроїв. Додатково розглянуто частковий випадок складання розкладу для декількох ідентичних пристроїв. Наведено приклад розв'язання задачі складання розкладу для декількох пропорційних пристроїв. Проведено серію експериментів для демонстрації залежності значення цільової функції від зміни директивного строку.

The article presents the problem of scheduling theory, in which it is necessary to schedule the tasks with a common due date by parallel machines on the criterion of minimizing the total deviation of the completion of tasks from the due date. Taking into account the conditions of polynomial and non-deterministic polynomial problem solvability, algorithms scheduling for several proportional machines has been developed. Partial cases of scheduling for several identical machines are additionally considered. An example of scheduling task for several proportional machines is given. A series of experiments has been conducted to demonstrate the dependence of the value of the target function on the change of the due date.

Ключові слова: РОЗКЛАД, ДИРЕКТИВНИЙ СТРОК, МОМЕНТ ЗАПУСКУ, ПАРАЛЕЛЬНІ ПРИСТРОЇ, ІДЕНТИЧНІ ПРИСТРОЇ, ПРОПОРЦІЙНІ ПРИСТРОЇ, ЗАПІЗНЕННЯ, ВИПЕРЕДЖЕННЯ, МІНІМІЗАЦІЯ СУМАРНОГО ВІДХИЛЕННЯ.

1. Постановка задачі

Задано множину завдань $I = \{1,2,..,i,.,n\}$ та кількість пристроїв m. Пристрої працюють паралельно, є взаємозамінними і відрізняються один від одного продуктивністю виконання завдань. Пристрої можна впорядкувати за швидкістю виконання завдання і цей порядок однаковий для всіх завдань. Для кожного пристрою j, $j = \overline{1,m}$ задано коефіцієнт продуктивності h_j такий, що тривалість виконання завдання i на пристрої j дорівнює $h_j p_i$. «Еталонним» будемо називати пристрій з коефіцієнтом продуктивності $h = 1$. У цьому випадку величина p_i є тривалістю виконання завдання i на еталонному пристрої. Величина h_j – коефіцієнт продуктивності пристрою j (якщо $h_j > 1$, то пристрій j менш продуктивний, ніж еталонний, якщо $h_j < 1$ – більш продуктивний).

Передбачається, що всі завдання множини I надходять одночасно в нульовий момент часу і мають спільний директивний строк d, процес обслуговування кожного завдання проходить без переривань до завершення обслуговування. Пристрої можуть починати свою роботу в ненульовий момент часу. Необхідно побудувати розклад, що мінімізує сумарне відхилення виконання завдань від директивного строку.

Введемо позначення:

C_i – момент закінчення завдання i;

$Z_i = \max\{0,\ C_i - d\}$ – запізнення завдання;

$E_i = \max\{0,\ d - C_i\}$ – випередження завдання; $Z = \sum_{i=1}^{n} Z_i$ – сумарне запізнення;

$E = \sum_{i=1}^{n} E_i$ – сумарне випередження;

$u = d - \sum_{i \in W} p_i$ – момент запуску розкладу (момент початку виконання найпершого із завдань).

З урахуванням позначень, цільова функція задачі (сумарне відхилення моментів

закінчення завдань від директивного строку) має вигляд: $E+Z\to\min$.

2. Складання розкладу для декількох ідентичних пристроїв

Нехай маємо деякий розклад виконання завдань, коли в системі є $m>1$ ідентичних пристроїв.

Введемо позначення: Q_j – множина завдань, що запізнюються на пристрої j, $|Q_j|=q_j$, $j=\overline{1,m}$; $j\lfloor i\rfloor$ – номер завдання, що виконується на пристрої j та є i-тим за порядком в послідовності завдань, що запізнюються на цьому пристрої; W_j – множина завдань, що на пристрої j виконуються з випередженням, $|W_j|=w_j$, $j=\overline{1,m}$ (завдання, у якого на пристрої j випередження дорівнює 0, будемо відносити до множини W_j); $j\lceil i\rceil$ – номер завдання, що є i-тим з кінця в упорядкуванні завдань з множини W_j (на пристрої j є i-тим в послідовності завдань множини W_j, якщо рахувати від директивного строку справа наліво по часовій осі).

З урахуванням введених позначень визначимо сумарне відхилення моментів закінчення від директивного строку (суму значень випереджень та запізнень) [1]:

$$E+Z=\sum_{j=1}^{m}\sum_{i=1}^{w_j}E_{j\lceil i\rceil}+\sum_{j=1}^{m}\sum_{i=1}^{q_j}Z_{j\lfloor i\rfloor}$$

Сумарне випередження:

$$E=\sum_{j=1}^{m}\sum_{i=1}^{w_j}E_{j\lceil i\rceil}=\sum_{j=1}^{m}\sum_{i=1}^{w_j-1}(w_j-i)\,p_{j\lceil i\rceil}.$$

Сума $\sum_{j=1}^{m}\sum_{i=1}^{w_j-1}(w_j-i)\,p_{j\lceil i\rceil}$ не містить доданків, що відповідають завданням $p_{j\lceil w_j\rceil}$ (оскільки їх тривалості не впливають на значення $E_{j\lceil 1\rceil},E_{j\lceil 2\rceil},\ldots,E_{j\lceil w_j\rceil}$), введемо в ці суми доданки, що відповідають $p_{j\lceil w_j\rceil}$, з коефіцієнтами 0: $\sum_{j=1}^{m}\sum_{i=1}^{w_j}(w_j-i)\,p_{j\lceil i\rceil}$

Сумарне запізнення:

$$Z=\sum_{j=1}^{m}\sum_{i=1}^{q_j}Z_{j\lfloor i\rfloor}=\sum_{j=1}^{m}\sum_{i=1}^{q_j}(q_j-i+1)\,p_{j\lfloor i\rfloor}.$$

Сумарне відхилення:

$$E+Z=\sum_{j=1}^{m}\sum_{i=1}^{w_j}(w_j-i)\,p_{j\lceil i\rceil}+\sum_{j=1}^{m}\sum_{i=1}^{q_j}(q_j-i+1)\,p_{j\lfloor i\rfloor}.$$

3. Умови поліноміальної розв'язуваності задачі з ідентичними пристроями

1) Достатньо велике значення директивного строку

Згідно [2] існує процедура поліноміального розв'язання за умови:

$$d\ge\Delta=p_n+p_{n-2m}+\ldots+p_{n-2m\lceil(n-2m)/2m\rceil},\quad(1)$$

де Δ – сума тривалостей завдань, які за поліноміальним алгоритмом потрапляють до множини випереджаючих на першому пристрої.

При виконанні умови поліноміальної розв'язуваності (1), для мінімізації сумарного відхилення необхідно розподіляти завдання між пристроями за правилом «найдовшому завданню повинен відповідати найменший коефіцієнт ЦФ».

Алгоритм № 1

КРОК 1. Впорядкувати завдання за незростанням тривалостей: $p_1\ge p_2\ge\ldots\ge p_n$;

КРОК 2. Впорядкувати значення коефіцієнтів ЦФ за зростанням;

КРОК 3. ЦИКЛ по завданням *i* від 1 до *n*;

Завдання i призначити на пристрій з найменшим вільним коефіцієнтом ЦФ (який визначає позицію завдання в упорядкуванні відповідного пристрою).

Згідно Алгоритму № 1, спочатку заповнюються перші позиції всіх пристроїв (їм відповідають коефіцієнти 0), потім останні позиції всіх пристроїв (їм відповідають коефіцієнти 1), після цього – другі позиції всіх пристроїв (їм відповідають коефіцієнти 1) і т.д.

2) Покращення критерію загального часу виконання завдань

Введемо позначення

$$F=\max_{1\le j\le m}\{\sum_{i=1}^{w}p_{j\lceil w\rceil}\}+\max_{1\le j\le m}\{\sum_{i=1}^{q}p_{j\lceil q\rceil}\},$$

де F – загальний час виконання завдань у розкладі.

Значення критерію можна покращити, якщо поміняти місцями значення завдань $p_{j\lceil w\rceil}$ та $p_{j\lceil q\rceil}$ на першому і останньому, другому і передостанньому, і т.д. пристроях. Описані дії можна виконувати, бо вони не вплинуть на значення цільової функції.

3) Достатньо мале значення директивного строку

Якщо директивний строк достатньо малий, а саме:

$$d < \min_{1\le i\le n}\{p_i\}, \qquad (2)$$

то у будь-якому розкладі усі завдання запізнюються, тоді SPT- упорядкування дає оптимальний розклад [3].

4. Умови неполіноміальної розв'язуваності задачі з ідентичними пристроями

При відсутності достатнього люфту при складанні розкладу, тобто при невиконанні умови (1), задача відноситься до класу NP.

В основу розробленого алгоритму покладено алгоритм розв'язання задачі при виконанні умови (1). Попередньо упорядковуються завдання за незростанням. Потім за Алгоритмом № 1, при визначенні позиції завдання у розкладі, враховується яким чином це вплине на допустимість розкладу. Алгоритм № 1 виконується до тих пір, доки тривалість поточного завдання не почне перевищувати проміжок часу, що залишився між моментом початку виконання завдання та директивним строком.

В такому випадку крок 3 Алгоритму № 1 модифікується таким чином: при визначенні місця поточного завдання виконується порівняння значення його тривалості з проміжком часу, що залишився між можливим моментом початку виконання завдання та директивним строком.

Алгоритм № 2

КРОК 1. Впорядкувати завдання за незростанням тривалостей: $p_1 \ge p_2 \ge \ldots \ge p_n$;

КРОК 2. Впорядкувати значення коефіцієнтів ЦФ за зростанням;

КРОК 3. ЦИКЛ по завданням *i* від 1 до *n*;

ЯКЩО завдання *i* (завдання з поточної множини непризначених завдань) має максимальну тривалість;

ТА $p_i \le (d - \sum_{j=1}^{m}\sum_{k=1}^{w_j} p_{\lceil k\rceil})$;

ТОДІ призначити завдання до множини W на пристрій з найменшим вільним коефіцієнтом ЦФ;

ІНАКШЕ призначити завдання до множини Q на пристрій з найменшим вільним коефіцієнтом ЦФ (який визначає позицію завдання в упорядкуванні відповідного пристрою).

Згідно алгоритму № 2 спочатку заповнюється перша позиція пристрою, потім остання.

5. Складання розкладу для декількох пропорційних пристроїв

1) Достатньо велике значення директивного строку

У цьому випадку оцінити в закритій формі (подібній формулі (1)) максимальний сумарний час тривалостей завдань, що йдуть с випередженням на пристроях неможливо.

Спочатку розглянемо поліноміальний алгоритм побудови оптимального розкладу завдань за умови, коли d є достатньо великим числом, тобто таким, що у будь-якому розкладі $u \ge 0$. Визначимо для випадку паралельних пропорційних пристроїв сумарне відхилення моментів закінчення від директивного строку [1]:

Сумарне випередження:

$$E = \sum_{j=1}^{m}\sum_{i=1}^{w_j} E_{j\lceil i\rceil} = \sum_{j=1}^{m} h_j \sum_{i=1}^{w_j-1}(w_j - i)\,p_{j\lceil i\rceil}$$

Сумарне запізнення:

$$Z = \sum_{j=1}^{m}\sum_{i=1}^{q_j} Z_{j\lfloor i\rfloor} = \sum_{j=1}^{m} h_j \sum_{i=1}^{q_j}(q_j - i + 1)\,p_{j\lfloor i\rfloor}.$$

Сумарне відхилення:

$$E + Z = \sum_{j=1}^{m} h_j \sum_{i=1}^{w_j}(w_j - i)\,p_{j\lceil i\rceil} +$$

$$+ \sum_{j=1}^{m} h_j \sum_{i=1}^{q_j}(q_j - i + 1)\,p_{j\lfloor i\rfloor}.$$

Отже, $E + Z$ являє собою суму n доданків ($n = \sum_{j=1}^{m}(w_j + q_j)$), кожний з яких є добутком коефіцієнту продуктивності j-го пристрою, цілого числа (що відповідає позиції завдання у розкладі відповідного пристрою) і тривалості завдання.

Назвемо складеним коефіцієнтом ЦФ добуток коефіцієнта продуктивності пристрою на ціле число, яке відповідає позиції завдання у розкладі.

Для мінімізації сумарного відхилення необхідно розподіляти завдання між пристроями за правилом «найдовшому завданню повинен відповідати найменший складений коефіцієнт ЦФ».

Отже пропонується вчинити наступним чином: за алгоритмом № 3 будується розклад, який перевіряється на допустимість.

Алгоритм № 3

КРОК 1. Впорядкувати завдання за незростанням тривалостей: $p_1 \geq p_2 \geq ... \geq p_n$;

КРОК 2. Сформувати можливі значення складених коефіцієнтів ЦФ та впорядкувати їх за зростанням;

КРОК 3. ЦИКЛ по завданням;

Завдання i (завдання з поточної множини непризначених завдань, яке має максимальну тривалість) призначити на ту позицію у розкладі, якій відповідає найменший вільний складений коефіцієнт.

Якщо розклад, побудований за Алгоритмом № 3, є допустимим (момент запуску розкладу невід'ємний) – то він є і оптимальним.

2) Достатньо мале значення директивного строку

Якщо директивний строк достатньо малий, а саме:

$$d < \min_{1 \leq i \leq n}\{p_i\} \cdot \min_{1 \leq j \leq m}\{h_j\}, \qquad (3)$$

то у будь-якому розкладі усі завдання запізнюються, тоді SPT- упорядкування дає оптимальний розклад.

3) Загальний випадок задачі

В основу розробленого алгоритму покладено алгоритм № 3. Попередньо упорядковуються завдання за незростанням тривалостей, формуються можливі значення складених коефіцієнтів ЦФ та впорядковуються за зростанням. Потім за Алгоритмом № 3, при визначенні позиції завдання у розкладі, враховується яким чином це вплине на допустимість розкладу. Алгоритм № 3 виконується до тих пір, доки тривалість поточного завдання з урахуванням коефіцієнту продуктивності не почне перевищувати проміжок часу, що залишився між моментом початку виконання завдання та директивним строком.

В такому випадку крок 3 Алгоритму № 3 модифікується таким чином: при визначенні місця поточного завдання виконується порівняння значення його тривалості з урахуванням коефіцієнту продуктивності з проміжком часу, що залишився між можливим моментом початку виконання завдання та директивним строком.

Алгоритм № 4

КРОК 1. Впорядкувати завдання за незростанням тривалостей: $p_1 \geq p_2 \geq ... \geq p_n$;

КРОК 2. Сформувати можливі значення складених коефіцієнтів ЦФ та впорядкувати їх за зростанням;

КРОК 3. ЦИКЛ по завданням i від 1 до n;

ЯКЩО завдання i (завдання з поточної множини непризначених завдань) має максимальну тривалість;

ТА найменший вільний складений коефіцієнт знаходиться на позиції l_w на пристрої з коефіцієнтом продуктивності h_j;

ТА $(p_i \cdot h_j) \leq (d - \sum_{j=1}^{m}\sum_{k=1}^{w_j} p_{\lceil k \rceil})$;

ТО призначити завдання до множини W на пристрій з найменшим вільним складеним коефіцієнтом;

ІНАКШЕ призначити завдання до множини Q на пристрій з найменшим вільним складеним коефіцієнтом (який визначає позицію завдання в упорядкуванні відповідного пристрою).

6. Експерименти

Для демонстрації роботи алгоритму складання розкладу для декількох пропорційних пристроїв розглянуто наступну задачу:

Задано $n = 25$ завдань і $m = 3$ пристроїв. Директивний строк $d = 65$. Коефіцієнти продуктивності пристроїв мають значення: $h_1 = 1,\ h_2 = 1{,}3,\ h_3 = 1{,}8$.

Тривалість виконання завдань на еталонному пристрої задано вектором:

$$p = \{20,18,16,13,12,12,11,11,10,10,$$
$$10,9,8,8,8,7,6,6,5,5,4,3,3,3,2\}.$$

Побудуємо розклад за алгоритмом № 3 і перевіримо, чи є отриманий розклад допустимим.

На рисунку 1 показано оптимальний розклад виконання завдань (у прямокутниках вказано номер завдання та тривалість його виконання на відповідному пристрої, над прямокутниками вказані значення складених коефіцієнтів.

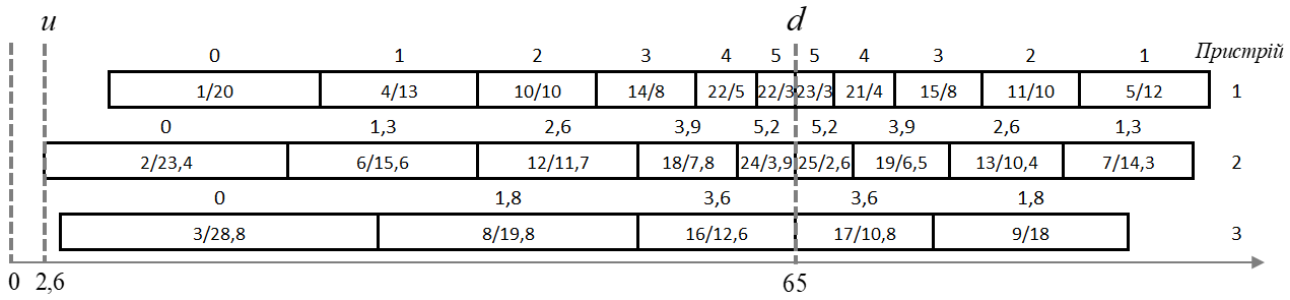


Рис. 1. Оптимальний розклад розглянутої задачі

Сумарне випередження:

$$E = 92 + 78 + 45 = 215.$$

Сумарне запізнення:

$$Z = 87 + 65 + 39{,}6 = 191{,}6.$$

Сумарне відхилення: $L = E + Z = 406{,}6$.

Момент запуску розкладу $u = 2{,}6$.

Оскільки момент запуску розкладу додатній, то отриманий розклад є оптимальним.

Проведемо серію експериментів для спостереження за зміною значення цільової функції в залежності від зменшення директивного строку, використовуючи Алгоритми № 3 та № 4.

В таблиці 1 наведено розрахунки значень сумарного випередження, сумарного запізнення, сумарного відхилення та моменту запуску розкладу отримані в результаті зменшення значення директивного строку на 3 одиниці в кожному новому експерименті.

Для наочності побудовано графік залежності цільової функції задачі від зміни директивного строку, що наведений на рисунку 2.

Згідно отриманих результатів можна зробити наступні висновки: використання розроблених алгоритмів дозволяє знайти оптимальний або наближений до оптимального розв'язок задачі, а також з'ясувати як зменшення директивного строку впливає на зміну значення цільової функції. Завдяки цьому можна знайти мінімально можливий директивний строк, за якого отриманий розклад буде залишатись оптимальним.

Табл. 1. Результати розрахунків

№	1	2	3	4	5	6	7	8	9	10	11
d	65	62	59	56	53	50	47	44	41	38	35
E	215	209,8	192,2	161,9	123	110,2	94,2	70,8	55,4	46,3	32,8
Z	191,6	196,8	215,1	250,5	312,8	333,4	357,4	409,5	448,9	471,8	517
E + *Z*	406,6	406,6	407,3	412,4	435,8	443,6	451,6	480,3	504,3	518,1	549,8
u	2,6	0,8	0	0	0,8	0	0	0,8	0	0	0
№	12	13	14	15	16	17	18	19	20	21	22
d	32	29	26	23	20	17	14	11	8	5	2
E	19,8	14,2	8,6	3	2,6	0	0	0	0	0	0
Z	573,6	595,6	630,8	675,1	690,1	721,7	746,1	767,8	811	878	959,7
E + *Z*	593,4	609,8	639,4	678,1	692,7	721,7	746,1	767,8	811	878	959,7
u	0	0	0	0	0	0,1	1	0	0	0	0

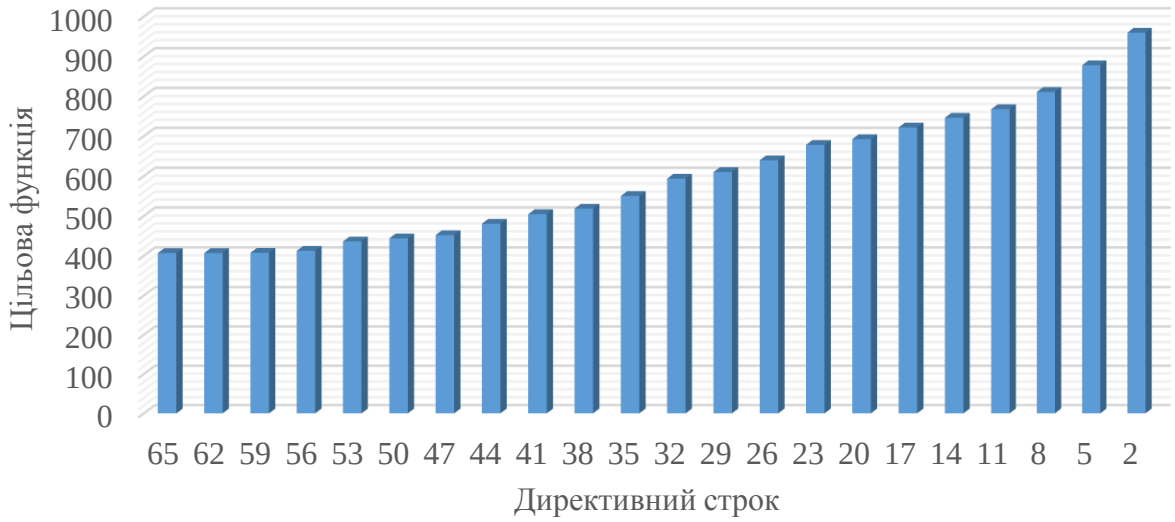


Рис. 2. Графік залежності сумарного відхилення від директивного строку

Перелік посилань

1. Задача мінімізації сумарного відхилення від спільного директивного строку при виконанні завдань паралельними пристроями / А. В. Годна, А. О. Маленко, О. Г. Жданова, М. О. Сперкач. // Науковий огляд. – 2017. – №9(14). – С. 14–32.
2. Ventura J. A. An improved dynamic programming algorithm for the single-machine mean absolute deviation problem with a restrictive common due date / J. A. Ventura, M. X. Weng. // Operations Research Letters. – 1995. – №17 (3). – P. 149–152.
3. Конвей Р. В. Теория расписаний / Р. В. Конвей, В. Л. Максвелл, Л. В. Миллер. – Москва: Наука. Главная редакция физ.-мат. литературы, 1975. – 360 с.

УДК 519.1

ПАНІЙВАН В. Ю.

СИСТЕМА ВИДАННЯ РЕКОМЕНДАЦІЙ ДЛЯ КЕРУВАННЯ ЧЕРГ У СИСТЕМАХ МАСОВОГО ОБСЛУГОВУВАННЯ

У даній статті описано проект створення системи контролю черг для многоканальних систем масового обслуговування (СМО) з необмеженим очікуванням. У рамках даної статті буде описано побудову моделі СМО для емуляції роботи гірськолижного курорту та створення системи видачі рекомендацій на основі характеристик та стану створеної моделі. Основною ціллю проекту є створення універсальної системи видачі рекомендацій щодо обрання черги для зменшення часу очікування.

This article describes the project of creating a system of queue control for multi-channel mass public service system (MPSS) with unlimited expectation time. Within the framework of this article will be described the construction of the MPSS model for emulating the work of the ski resort and the creation of a system for issuing recommendations based on the characteristics and state of the created model. The main goal of the project is to create a universal system for issuing recommendations for choosing a queue to reduce waiting time.

1. Вступ

Проблема керування чергами гостро стоїть у всіх системах масового обслуговування (СМО). Ціллю даної статті є опис проекту створення системи контролю людського потоку для многоканальних СМО з необмеженим очікуванням. Прикладом подібних систем можуть бути парки атракціонів або гірськолижні куроти. У рамках даної статті буде описано побудову моделі СМО для емуляції роботи гірськолижного курорту та створення системи видачі рекомендацій на основі характеристик та стану створеної моделі. Основною ціллю проекту є мінімізація часу очікування між катаннями.

2. Побудова моделі

Систему гірськолижних трас можна представити у вигляді графу. Вершинами графу будуть точки підйому та спуску, тобто граф має два типи вершин: підйомники та траси. Обидва типи вершин мають якість наявності людей (черг). Ребрами графу є траси підйому та спуску, а їх вагами - час потрібний для спуску або підйому. Окрім черг, вершини графу мають ще додаткову характеристику у вигляді структури даних що відображає прогнозовану кількість людей, що прибуде на цю вершину, а також час через який це здійсниться. Нижче наведено тестовий приклад спроектованої системи та результати її роботи. На рисунку 1 представлена схема системи з трьох вершин.

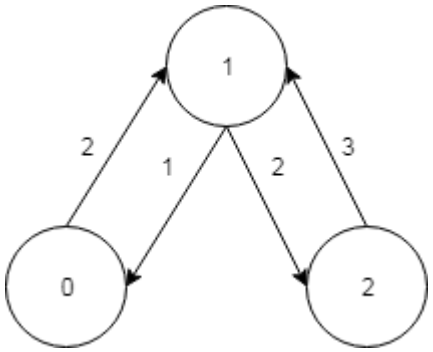


Рис. 1. Схема тестової системи.

У таблиці 1 описано статичні характеристики вершин графу.

Табл. 1. Статичні дані вершин графу.

№ вершиини	Тип вершини	Пропускна здатність чол/хв
0	Підйомник	10
1	Траса	-
2	Підйомник	20

У таблиці 2 описані змінні характеристики графу, такі як поточна довжина

черг та очікувана кількість людей, що прибудуть на підйомник у деякий час. Для наочності дані наведено для перших декількох хвилин роботи моделі з кроком в одну хвилину.

Табл. 2. Змінні характеристики вершин протягом роботи системи.

Хвилин з початку	№ вершини	Поточна черга	Прибуття людей очікується
0	0	200	-
	1	0	-
	2	300	-
1	0	190	-
	1	0	10 людей через 1 хвилину 20 людей через 2 хвилину
	2	280	-
2	0	180	-
	1	10	30 людей через 1 хвилину 20 людей через 2 хвилину
	2	260	-
3	0	175	-
	1	30	30 людей через 1 хвилину 20 людей через 2 хвилину
	2	240	5 людей через 1 хвилину
4	0	180	-
	1	30	30 людей через 1 хвилину 20 людей через 2 хвилину
	2	225	15 людей через 1 хвилину

Як можна бачити із таблиці, модель працює коректно, тому що кількість людей у системі залишається незмінною. У рамках даного прикладу для симуляції поведінки лижників був використаний закон рівномірного розподілу.

3. Видача рекомендацій

Для лижника що стоїть на вершині найкращим шляхом можна вважати той, що мінімізує час очікування після спуску, тобто призводить до найменших затримок між катаннями. Час очікування можна виразити наступною формулою:

$$t = q / p + v,$$

де t - час очікування, q - довжина черги на момент прибуття, p - пропускна здатність підйомника, v - час підйому підйомника. При цьому довжину черги можна виразити формулою:

$$q = q_0 - p * w + \sum_{i=1}^{n} q_i,$$

де q_0 - довжина черги на момент видачі рекомендації, w - тривалість спуску, q_i - довжина черги яка додасться до поточної через час w, n - кількість черг q_i які додадуться до поточної через час w.

4. Приклад роботи системи видачі рекомендацій

Для моделі що була описана у таблицях 1 - 2 та рис. 1 за результатом роботи протягом 4 хвилин система набула наступного вигляду вигляду (табл. 3):

Табл. 3. Стан системи після чотирьох хвилин роботи.

№ вершини	Поточна черга	Прибуття людей очікується
0	180	-
1	30	30 людей через 1 хвилину 20 людей через 2 хвилину
2	225	15 людей через 1 хвилину

Для лижника що знаходиться у вершині 1 постає задача вибору шляху спуску - до вершини 0 чи до вершини 2. У даному випадку найкоротший час очікування буде досягнутий при виборі траси до вершини 2 (13.375 хвилин у

порівнянні з 19.75 хвилинами), незважаючи на те що черга у вершині 2 набагато довша.

5. Вплив системи видачі рекомендацій на середній час очікування

Розглянемо наступний приклад (рис. 2).

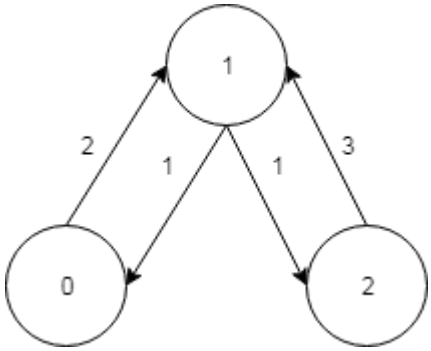


Рис. 2. Схема тестової системи.

У таблиці 4 описано статичні характеристики вершин графу.

Табл. 4. Статичні дані вершин графу.

№ вершиини	Тип вершини	Пропускна здатність (чол/хв)
0	Підйомник	2
1	Траса	-
2	Підйомник	3

Нехай дані з вершин мають наступний вигляд (табл. 5):

Табл. 5. Стан системи.

№ вершини	Поточна черга	Прибуття людей очікується
0	8	-
1	5	3 людини через 1 хвилину
2	12	-

При умові що усі лижники користуються системою видачі рекомендацій, 3 з лижників, що стоять на вершині 1 поїдуть до 2 вершини, інші 2 - до 0. Вигляд системи буде таким самим як на табл. 5. При цьому середній час очікування для цих п'яти лижників буде дорівнювати рівно 4 хвилини. Тепер розглянемо приклад коли один з лижників зневажив рекомендацією та поїхав на вершину 0 замість вершини 2. Система набуде наступного вигляду (табл. 6.):

Табл. 6. Стан системи у разі зневаження рекомендацією.

№ вершини	Поточна черга	Прибуття людей очікується
0	9	-
1	5	3 людини через 1 хвилину
2	11	-

При цьому середній час очікування для цих п'яти лижників збільшиться з 4 хвилин до 4.164 хвилин.

6. Висновок

На практиці дані в подібній системі мають бути доповнені даними з зовнішніх систем спостереження для створення більш чіткої моделі. Також модель може бути ускладнена додатковими параметрами, такими як ввід-вивід людей із системи, зміна часу спуску в залежності від навичок лижника та інші. Використання подібної системи не обмежено гірськолижними курортами та може бути застосовано для будь-яких СМО.

Список літератури

1. Баскетт, Ф., Ченді Л., Мані К., Мантц Р.Р., Паласіос Ф.Г.. Відкриті, закриті та змішані мережі черг різних класів клієнтів // Журнал ACM, 1975. - 248–260с.
2. Брамсон М. Стабільна мережа черг з нестабільною моделлю потоку, 1999.
3. Сандаранпандіан, В. Теорія масового обслуговування, 2009.

УДК 004.043

СТОРЧЕВИЙ В.В.,
ЖДАНОВА О.Г.

ЗАСТОСУВАННЯ МЕТОДУ ЛОКАЛЬНО-ЧУТЛИВОГО ХЕШУВАННЯ ДЛЯ ВИРІШЕННЯ ЗАДАЧІ ЗНАХОДЖЕННЯ ПОДІБНИХ ЕЛЕМЕНТІВ, ЗАДАНИХ МНОЖИНОЮ ХАРАКТЕРИСТИК

Пропонується підхід, який дозволяє знаходити схожих користувачів у системі з великою кількістю даних. Цей підхід включає використання хешування та імовірнісні методи зниження розмірності багатовимірних даних. Визначено метод перевірки елементів на схожість на основі створення сигнатур за допомогою сімейства хеш-функцій для міри Жаккара, яке носить назву MinHash. Проведено дослідження імовірності знаходження пар схожих множин при різних значеннях коефіцієнта Жаккара для них.

This article proposes an approach that allows you to find similar users in a system with a large amount of data, where it is inappropriate to use clustering methods. This approach is based on the use of hashing and probabilistic methods for reducing the dimension of multidimensional data. Defined method of checking the similarities based on creating signatures using the family of hash functions for Jaccard measure, which is called MinHash. Studies have been made of the probability of finding pairs of similar sets at different values of the Jaccard coefficient for them.

Ключові слова: РЕКОМЕНДАЦІЙНА СИСТЕМА, ХЕШ-ФУНКЦІЯ, MINHASH, ЛОКАЛЬНО-ЧУТЛИВЕ ХЕШУВАННЯ, ПОДІБНІСТЬ КОРИСТУВАЧІВ

1. Вступ

На сьогоднішній день рекомендаційні системи є досить актуальними. Вони допомагають користувачам знаходити певні товари, інформацію, людей, які в деякому сенсі будуть для них найбільш цікавими, не витрачаючи на це багато часу. Для створення рекомендацій можуть використовуватися дані про дії користувача або його характеристики. У даній роботі розглядається задача знаходження подібних користувачів, тобто користувачів зі схожими вподобаннями. На сьогодні задача знаходження подібних елементів – одна з ключових проблем інтелектуального аналізу даних. Вирішення даної задачі є тривіальним, коли йде мова про невеликі об'єми даних.

Для вирішення даної задачі використовуються наступні методи: сингулярний розклад матриці (SVD); пошук асоціативних правил; кластеризація.

Метод сингулярного розкладу матриці базується на зменшенні розмірів задачі. Він надає високоякісні рекомендації, але використовує матричні розрахунки, які потребують багато обчислювального часу [1].

Розробка правил асоціації - один з найважливіших прийомів глибинного аналізу даних, він шукає закономірності між предметами у заданому наборі даних. Асоціативний аналіз - це виявлення правил асоціації атрибут-значення, які часто зустрічаються разом у певному наборі даних [2]. Кластеризація - об'єднання в групи схожих об'єктів - є однією з фундаментальних задач в області аналізу даних [3].

Дані методи є неефективними для великих та надвеликих масивів даних. Для роботи з ними на допомогу приходять імовірнісні методи зниження розмірності багатовимірних даних.

2. Постановка задачі

2.1 Визначення схожості двох наборів даних

Коефіцієнт Жаккара – перша міра подібності, запропонована Полем Жаккаром у 1901 році [4]. Нехай A та B – користувачі

деякої системи і їх вподобання представлені у вигляді множин S_A та S_B. Коефіцієнт Жаккара визначається за формулою:

$$J(S_A, S_B) = \frac{|S_A \cap S_B|}{|S_A \cup S_B|}.$$

За визначенням $0 \le J(S_A, S_B) \le 1$, ідентичні набори будуть мати коефіцієнт рівний одиниці, абсолютно різні набори – нулю.

Приклад 1. Нехай S_A= [футбол, теніс, хокей, танці, подорожі]; S_B= [книги, теніс, подорожі]. На перетині маємо 2 елементи (теніс, подорожі), а в об'єднанні – 6 (футбол, теніс, книги, хокей, танці, подорожі). Тоді відповідний коефіцієнт такий:

$$J(S_A, S_B) = \frac{2}{6} = \frac{1}{3}.$$

2.2 Постановка задачі знаходження подібних елементів за їх множинами характеристик

Дано: загальна кількість користувачів (m); множини вподобань користувачів $S_1, ..., S_m$ (кожна з них представляє собою деяку множину рядків); кількість кошиків, у які будуть розподілені користувачі (k) - кількість множин користувачів, подібних між собою.

Необхідно розбити множину користувачів на k підмножин (кошиків) за їх подібністю по метриці Жаккара, спираючись на їх вподобання.

3. Застосування сигнатур

Якщо вподобання користувача представлені у вигляді множини рядків (а це зазвичай має місце в рекомендаційних системах), то їх збереження вимагає багато місця в пам'яті. Наша мета полягає в тому, щоб замінити великі набори на набагато менші представлення, які будемо називати сигнатурами. Сигнатура – це результат обчислення хеш-функції. Процес перетворення множин у сигнатури називається мінхешингом (MinHash) [5]. Важлива властивість сигнатури полягає в тому, що ми можемо зіставити сигнатури двох наборів і оцінити коефіцієнт Жаккара лише на їх основі [6].

Загальна схема оцінювання схожості складається з трьох етапів:

ЕТАП I. Побудова матриці характеристик.

ЕТАП II. Побудова матриці сигнатур.

ЕТАП III. Локально-чутливе хешування.

Розглянемо ці етапи та продемонструємо перші два на даних *Прикладу 1*.

Нехай $U = \bigcup_{i=1}^{m} S_i$ - універсальна множина вподобань в системі, що розглядається. Множині S_i може бути поставлено у відповідність булевий вектор довжини $|U|$, у якому 1 означає, що відповідний елемент з універсальної множини присутній у S_i, і 0 означає відсутність цього елементу в S_i. З кінцевим числом підмножин з U можна пов'язати *матрицю характеристик*, де кожен стовпець представляє собою вектор, що відповідає певній підмножині і кожен рядок відповідає елементу U [7].

Побудуємо матрицю M характеристик для наборів даних S_A та S_B.

Табл. 1. Матриця характеристик для множин S_A та S_B з Прикладу 1

Характеристика	**Елементи матриці**	
	S_A	S_B
Футбол	1	0
Теніс	1	1
Книги	0	1
Хокей	1	0
Танці	1	0
Подорожі	1	1

У таблиці 1 стовпці відповідають множинам S_A та S_B, що представляють вподобання користувачів, та універсальна множина U, що складається з шести варіантів вподобань.

На етапі II створюється матриця сигнатур. Для створення цієї матриці необхідно підібрати n випадкових хеш-функцій, n підбирається емпірично, чим воно більше, тим більша точність алгоритму, але також і більший час роботи.

Для цього використаємо наступний алгоритм.

Алгоритм створення матриці сигнатур

Вхідні дані:

- n хеш-функцій $h_1, ...h_n$;
- M – матриця характеристик.

КРОК 1 Створити матрицю H зі n рядків із $|S|$ стовпців для зберігання матриці і присвоїти усім елементам значення ∞.

КРОК 2 ЦИКЛ по рядках матриці M: для r-го рядку виконати наступні кроки:

2.1 Вирахувати значення $h_1(r),...h_n(r)$.

2.2 ЦИКЛ по стовпцях матриці M: для c-го стовпця виконати: якщо $M[r,c]=1$, то $H[i,c] := \min(H[i,c], h_i(r))$ для $i=1,...,n$.

Відмітимо, що для кожного ненульового елементу матриці M буде проведено $O(n)$ обчислень.

Виконаємо мінхешинг для даних *Прикладу 1*.

Нехай маємо $n=2$ хеш функції

$h_1(r) = r \bmod 6$,

$h_2(r) = (2r+1) \bmod 6$

На основі матриці характеристик створимо матрицю сигнатур, що представлена в табл. 2.

Табл. 2. Матриця H сигнатур для множин S_A та S_B з Прикладу 1

Хеш-функція	**Значення хеш-функцій для множин**	
	S_A	S_B
h_1	2	2
h_2	0	1

Як бачимо, коефіцієнт подібності дорівнює 0.5. Він відрізняється від коефіцієнта Жаккара, який ми обраховували раніше. Для того, щоб отримати більш точний результат, треба збільшити кількість функцій n.

Матриця сигнатур, отримана в результаті мінхешингу, має менший розмір у порівнянні з матрицею характеристик. Може бути так, що її розмір все-одно буде досить великим. До того, якщо усі множини повинні порівнюватись попарно, то для великої кількості даних таке «пряме» порівняння займе багато часу. Для вирішення цієї проблеми будемо використовувати імовірнісний метод зниження розмірності багатовимірних даних - так зване локально-чутливе хешування (LSH).

4. Локально-чутливе хешування

Локально-чутливе хешування – загальна назва методів, що дозволяють отримати подібні сигнатури для схожих наборів даних. Метод орієнтується на локальну чутливість, тобто при зміні якоїсь частини об'єкта, що хешується, сам хеш не повинен абсолютно змінюватися [8]. Основна ідея полягає в такому підборі хеш-функцій для деяких вимірювань, щоб схожі об'єкти з високим ступенем ймовірності потрапляли в один кошик.

Множини повинні багато разів хешуватися, це значно збільшує імовірність збігів або відмінностей. Лише множини, що потрапили до одного кошику, будуть у подальшому розглядатися як потенційно схожі. Якщо ж дві множини лежать у різних кошиках, то вони більше не будуть порівнюватися між собою. Це не означає, що вони точно не схожі, імовірність цієї події залежить від параметрів методу [9].

Необхідно мінімізувати кількість відмінностей між множинами, що лежать в одній корзині.

Нехай ми маємо матрицю сигнатур H, тоді нам необхідно розділити її на b смуг, кожна з яких має r рядків.

Для кожної смуги визначаємо хеш-функцію $h: \mathbb{Z}^r \to \mathbb{Z}$, яка вектор-стовпець довжини r відображає у ціле число (номер кошику) [10].

Точність методу залежить від того, як ми підберемо параметри b та r.

Проведемо оцінку ефективності методу. Нехай у нас є дві множини S_1 та S_2 і коефіцієнт Жаккара для них становить 0.8. Припустимо, що кількість смуг $b=20$, кількість рядків у смузі $r=5$. Тоді ймовірність того, що S_1 та S_2 однакові в одному кошику, становить

$$P_{S_1 \& S_2} = 0.8^5 = 0.328$$

Імовірність того, що S_1 та S_2 не потрапили ні в один кошик:

$$P = (1-0.328)^{20} = 0.00035$$

Як бачимо, ми знайдемо 99.965% (як імовірність протилежної події) пар подібних множин.

Імовірність того, що сигнатури співпадуть хоча б у одній із груп [10], становить:

$P = 1-(1-s^r)^b$, де s – це коефіцієнт Жаккара для порівнюваних множин.

5. Результати експериментів

Метою проведення експериментів є дослідження того, як ступінь подібності вхідних множин впливає на їх групування по

кошикам. На кожному кроці для кожної величини подібності між парами множин (в діапазоні $[0,\ 0.1,\ 0.2,\ ...\ 1.0]$) була обчислена імовірність того, що ці дві пари потрапляють в один і той же кошик щонайменше на одному етапі.

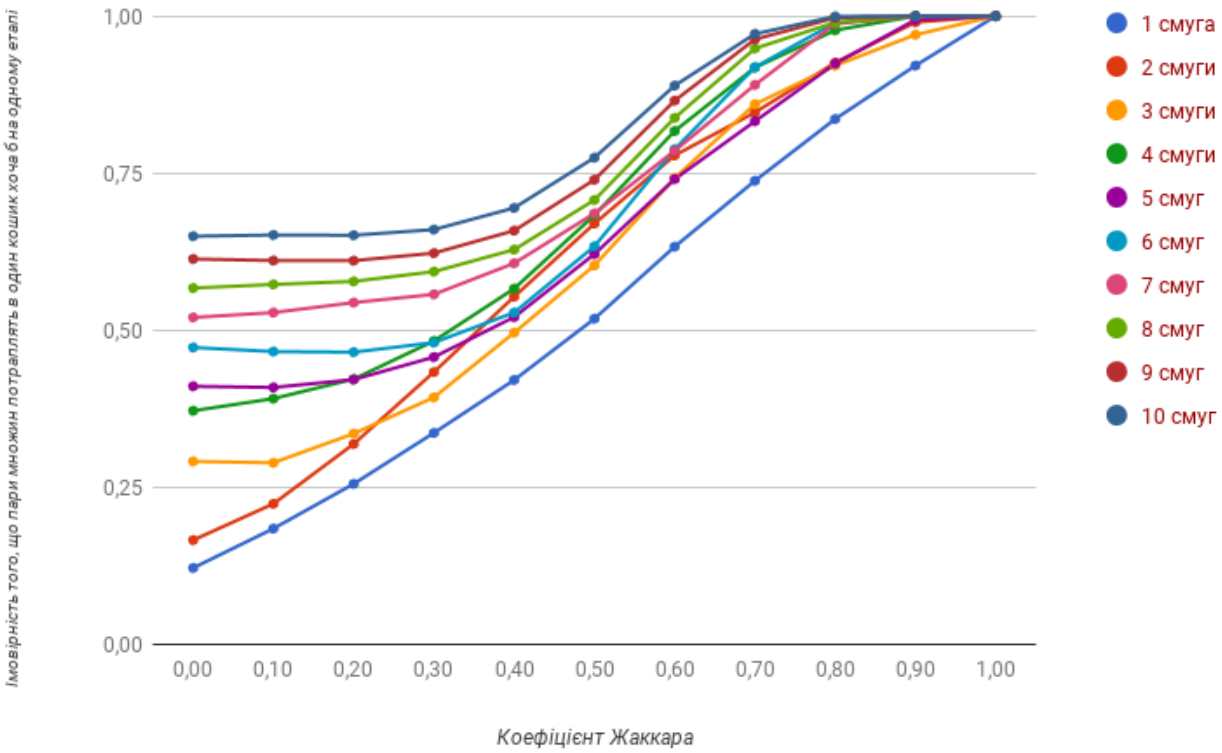


Рис. 1 – Залежність ймовірності того, що пари множин потраплять в один кошик хоча б на одному етапі, від коефіцієнту Жаккара

Різні серії представляють різні значення кількості смуг (від 1 до 10).

На рисунку можна розпізнати сигмоїду з порогом (точка, де крива найбільш крута) розташованим навколо $x = 0.5$. Ця крива є важливою, вона показує, що якщо коефіцієнт Жаккара буде вищим за 0.6, то скоріше за все майже всі множини потраплять у один кошик, усі інші ймовірно будуть пусті. З іншого боку, якщо множини відрізняються одна від одної (подібність нижче 0.2), то крива майже рівна. Це означає, що пара множин має однакову ймовірність потрапляння в один кошик незалежно від їх подібності.

6. Висновки

Як показали експерименти, локально-чутливе хешування - ефективний метод для роботи з даними великих розмірів, який за достатньо короткий час дозволяє знайти схожі елементи. Це досягається за допомогою локалізації множини елементів, тобто переглядаються тільки ті елементи, що мають імовірність бути схожими. Також за допомогою мінхешингу вдалося зменшити розмір вхідних даних, а також складність обрахунку коефіцієнта схожості, від $O(n^2)$ ми перейшли до $O(n)$.

Список літератури

1. Osman Nuri Osmanli, A Singular Value Decomposition Approach For Recommendation Systems, 2010.
2. Han J., Kamber M., Data Mining: Concepts and Techniques, Morgan Kaufman, 2001.
3. Jaccard P. Distribution de la flore alpine dans le Bassin des Dranses et dans quelques regions voisines // Bull. Soc. Vaudoise sci. Natur. — 1901. — V. 37, Bd. 140. — S. 241–272.
4. Sreenivas Gollapudi and Rina Panigrahy, The power of two min-hashes in similarity search among hierarchical data objects, PODS 2008: 211-220.
5. Пєчкурова О. М., Локально-чутливе хешування. Та сфери його застосування, 2017.
6. LSH Algorithm and Implementation (E2LSH) [Електронний ресурс]. – Режим доступу: http://web.mit.edu/andoni/www/ LSH/index.html. – LSH Algorithm and Implementation (E2LSH).
7. Фаустов Б. А., Поиск похожих изображений, Санкт-Петербургский Государственный Университет, 2016.
8. Anand Rajaraman Mining of Massive Datasets / Anand Rajaraman, Jeffrey David Ullman // Cambridge University Press, 2011.
9. Andoni A. Near-Optimal Hashing Algorithms for Approximate Nearest Neighbor in High Dimensions [Електронний ресурс] / Alexandr Andoni, Piotr Indyk. – Режим доступу: http://web. mit.edu/andoni/www/LSH/index.html. - LSH Algorithm and Implementation (E2LSH).
10. Anand Rajaraman Near Neighbor Search in High Dimensional Data [Електронний ресурс]. – Режим доступу: https://web.stanford.edu/class/cs345a/slides/05-LSH.pdf.

УДК 004.042; 004.056

СЯГАЙЛО Т.А.,
ЖАРІКОВ Е.В.,

УПРАВЛІННЯ ПРОЦЕСОМ ЗБЕРЕЖЕННЯ ДАНИХ В ГІПЕРКОНВЕРГЕНТНИХ СИСТЕМАХ

В роботі представлені моделі направлені на управління процесом роботи з даних в центрах обробки даних за допомогою використання міграцій та реплікацій даних між пристроями та серверами відповідно. Виконання міграції та реплікації даних відбувається в залежності поточних показників фізичних серверів та пристроїв

In the paper was presented models for managing processes of data saving within data centers with data migration and data replication between drivers and physical servers. Data migration and replication are dependent on current criteria of physical servers and drivers.

Ключові слова: гіперконвергентні системи, сховища, міграція даних, реплікація даних, збереження даних, пристрої збереження даних, фізичні сервери

1. Вступ

Вивчаючи на сьогодні об’єм даних, яку необхідно зберігати на записуючих пристроях постає проблема керування процесом збереження даних. На даний момент існують багато способів роботи зі сховищами, а саме:

− використання існуючих хмар, таких як GoogleDrive або OneDrive, які надають можливість зберігати обраний об’єм даних на їх серверах в залежності від ціни;

− використання таких технологій, як SAN (storage area network, перек. - система сховищ зберігання даних), NAS (network-attached storage, перек.-мережі зберігання даних) або DAS (direct-attached storage, перек. - сховища з прямим підключенням);

− налаштування власних сховищ.

Так як кожен з пристроїв має свої характеристики, такі як загальний об’єм даних, пропускна спроможність пристроїв та інші, то при налаштувані сховищ варто звертати увагу саме на них. Пристрої, які в найчастіше за все, використовують для збереження даних на сховищах це SSD(state-solid drive) та HDD (hard disk drive). Якщо розглядати дані пристрої, то перші звичайно мають кращі показники продуктивності, але їх об’єм є не настільки великим, а вартість є вищою, через це для зберігання великого об’єму даних, які не використовуються часто іноді обирають пристрої HDD. Хоча вже на сьогодні можна побачити, що ємність пристроїв SSD намагаються зробити більшою, так само як і продуктивність HDD пристроїв постійно покращуються з кожним роком. Але зараз в середньому вартість за 1GB HDD становить 6 центів в той час як 1 GB SSD – 17 центів [1]. Через це на сьогодні є поширеним налаштування гібридних сховищ, які складаються як з пристроїв SSD, так і з HDD.

Узагальнюючи дані види сховищ, називають багаторівневими, в них відбувається поділ пристроїв на рівні в залежності від продуктивності. Поширеним варіантом є поділ на три рівні [2], такі як:

− суперпродуктивні - продуктивність, яких є найбільш високою порівняно з іншими. На них зберігаються дані, які використовуються майже постійно;

− звичайні - є гіршими за попередні, але не дуже повільними, на них варто зберігати дані, які менше використовуються;

− повільні - мають найгірші показники продуктивності порівняно з попередніми, можуть слугувати для збереження копій даних сховища.

2. Дані сховищ

Запускаючи віртувальні машини на серверах [3], не відбувається постійного використання усіх файлів сховища. Загалом, з проведених авторами досліджень виявлено, що відсоткова кількість файлів, яка використовується для проведення 90 відсотків операцій в системі, використроє тільки 20 відсотків даних зі сховищ [4]. З чого можно зробити висновок, необхідності збереження інших 80 відсотків файлів на більш продуктивних пристроях. Наприклад, для налаштування Windows 7, розмір необхідного дискового простору складатиме

до 20 ГБ. Розмір дискового простору для всіх інших файлів складатиме 40 ГБ. При аналізувані файлів, які використовуються при роботі з операційною системою за допомогою програми Process Manager, було виявлено, що в середньому протягом декількох годин роботи віртуальної машини використовується до 1 ГБ системних файлів операційної системи, та в залежності від її завантаженості об'єм файлів, які використовуються будуть змінюватись. Результати показали, що майже всі системні файли, які використовуються операційною системою не змінюють розмір.

Отримані дані використовуються для конфігурації віртуальних машин та розміщення файлів між рівнями в залежності від необхідності їх використання.

3. Рішення щодо збереження даних на сховищах

Міграція даних. Так як доступ до даних на сховищах відбувається не рівномірно, то зберігати всі дані серверів на більш швидки пристроях стає не вигідним. Через це стає за ціль міграція даних між пристроями в залежності від їх поточної необхідності для віртуальних машин. Міграція повина відбуватись в односторонньому напряму з більш повільних пристроїв на більш швидкі, з урахуванням того, що з повільних пристроїв блоки не видаляються, а залишаються. Через це у випадку, коли дані стають непотрібними виникає необхідність їх видалення з більш швидких пристроїв з наявною копією на пристроях, які є повільнішими.

Одною з головних умов для міграції даних між пристроями є кількість доступу до блоків протягом деякого заданого проміжку часу, через це стає необхідність максимізації середнього значення доступу на пристроях SSD:

$$\max \frac{\sum_{j=1}^{t} q_j x_{kj}}{\sum_{j=1}^{t} x_{kj}}, (1)$$

де t –кількість блоків, які зберігаються на поточному фізичному сервері, q_j – кількість доступів до блоків протягом деякого часу, x_{kj} – розміщення блоків на пристроях SSD.

Цільова функція стає актуальною при виконані представлених нижче обмежень:

- на загальний об'єм даних, які збераються на пристрої не перевищують розмір пристрою:

для SSD пристроїв це :

$$\sum_{j=1}^{t} x_{kj} b < s_k, (2)$$

для HDD пристроїв це :

$$\sum_{j=1}^{t} x_{kj} b < h_k \quad (3)$$

В яких s_k – ємність k-го пристрою SSD, h_k - ємність k-го пристрою HDD, b – розмір одного блоку даних.

- на кожному рівні блок повинен зберігатись тільки в одному екземплярі:

для SSD пристроїв це :

$$\sum_{k=1}^{m} x_{kj} = 1, (4)$$

для HDD пристроїв це :

$$\sum_{k=1}^{c} x_{kj} = 1, (5)$$

в яких m – це кількість SSD пристроїв, а c - це кількість SSD пристроїв на фізичному сервері.

- кожен з блоків належить лише одному файлу:

$$\sum_{i=1}^{n} B_{ij} = 1, j = \overline{1,t}. \quad (6)$$

де B_{ij} –матриця належності блоків до файлів (1 – у випадку коли j-ий блок належить i-ому файлу, 0 – коли j-ий блок належить i-ому файлу).

Для виконання поставленої задачі є можливість використання алгоритмів локального пошуку, таких як генетичний алгоритм або алгоритм отжигу, які нададуть можливість пошуку оптимального розміщення блоків на пристроях SSD.

Реплікація. Ще одним способом управління даними є їх реплікація. В децентралізованих сховищах, тобто в тих яких нема одного місця збереження даних, а дані зберігаються на кожному фізичному сервері необхідність створення копій даних виникає через те, що не можно стовідсотково бути впевненим в збереженості даних. Вони можуть бути втрачені при наявності проблем

на фізичному сервері або у випадку виходу з ладу одного пристрою для збереження даних. Через це створення копій допоможе в уникненні проблем у втраті даних.

Протягом довгого часу роботи центрів обробки даних при збільшені кількості серверів або виходу з ладу інших, стає необхідним пошук нового серверу для збереження реплікацій. Через це необхідно знати поточні показники сервера для того, щоб обирати той, показники якого на даний момент стануть кращими для збереження. Показник серверу залежить від двох коефіцієнтів, одна з яких є наявність вільного місця для збереження реплікацій, інша це поточна завантаженість пристроїв серверу віртуальними машинами та реплікаціями з інших серверів.

Для знаходження першого коєфіцієнту, необхідно значення максимально вільного місця на серверах та значення вільного місця на поточному сервері:

$$wVM_l = \frac{\sum_{k=1}^{c} h_{lk} - \sum_{u=1}^{p} d_u R_{ul}}{\max\left(\sum_{k=1}^{c} h_{lk} - \sum_{u=1}^{p} d_u R_{ul}\right)}, (7)$$

в яких h_{lk}- ємність *k*-го пристрою *l*-го серверу, c - кількість HDD пристроїв на поточному сервері, d_u- розмір u-го файлу системи, p – кількість файлів системи, R_{ul}- матриця розміщення файлів на серверах включаючи реплікації.

Для знаходження другого коєфіцієнту:

$$wR_l = \frac{\sum_{k=1}^{c} w_{lk}}{\max \sum_{k=1}^{c} w_{lk}}, (8)$$

де w_{lk} – значення завантаженості пристрою на поточний момент.

Отже для пошуку місця для реплікацій даних необхідно:

$$\min\left(\frac{1}{wVm_l} + wR_l\right), l = \overline{1, z}. (9)$$

При умові виконання наступних обмежень:

- кількість даних на серверах не повині перевищувати загальну ємність серверів:

$$\sum_{u=1}^{p} d_u R_{ul} \leq \sum_{k=1}^{c} h_{lk}, (10)$$

- необхідна наявність реплікацій даних:

$$\sum_{l=1}^{z} R_{ul} > 1. (11)$$

Також варто звернути увагу, що повині виконуватись обмеження, які були описані в частині про міграцію.

Для реалізації даної задачі можно виконати алгоритм прямого пошуку, який буде знаходити кращі сервери для створення реплікацій на поточний момент.

4. Висновки

Під час управління процесами збереження даних необхідна наявність міграції або реплікації даних на сховищах, через те, слідкуючи за процесом розміщення можно зменшити час очікування доступу та уникнути можливість втрати необхідних даних.

В роботі представлено моделі, які направлені на покращення процесу доступу до даних в залежності від використовуваності поточних файлів та на пошук серверів-реплікантів, на яких необхідно створювати копії даних.

Також в роботі проведено аналіз використань даних віртуальних машин, який показав, необхідність використання багаторівневих сховищ для зменшення часу очікування даних на сховищах.

Список літератури

1. Consumer SSDs and hard drive prices are nearing parity. [Електроний ресурс] / Режим доступу: https://www.computerworld.com/article/3010395/solid-state-drives/consumer-ssds-and-hard-drive-prices-are-nearing-parity.html (дата звернення: 10.04.2018)
2. Архитектура многоуровневого хранения данных. [Електроний ресурс] / Режим доступу: http://www.ncm.ru/modelnyj-ryad/soft/point-storage-manager/tiered-storage.php (дата звернення: 10.04.2018)
3. E. Zharikov, O. Rolik and S. Telenyk, "An integrated approach to cloud data center resource management," *2017 4th International Scientific-Practical Conference Problems of Infocommunications. Science and Technology (PIC S&T)*, Kharkov, 2017, pp. 211-218.
4. What's Hot, What's Not: How Data Temperature Helps Manage Very Big Data. [Електроний ресурс] / Режим доступу: https://www.forbes.com/sites/teradata/2015/07/24/whats-hot-whats-not-how-data-temperature-helps-manage-very-big-data/#5ec2e7146ec6

УДК 004.94; 004.4; 004.62

ТЕРЕНТЬЄВ Р. А.,
ЖАРІКОВ Е.В.

ПРОГНОЗУВАННЯ ПОТРЕБИ РЕСУРСІВ СЕРВЕРНОЇ СИСТЕМИ В УМОВАХ ХМАРНИХ ОБЧИСЛЕНЬ

Запропоновано метод прогнозування потреби ресурсів серверної системи в умовах хмарних обчислень, який будує прогноз за допомогою кількох моделей прогнозування - ARIMA, ARIMAX, GMDH. При побудові прогнозу, метод обирає модель яка дала найточніший прогноз на попередньому кроці.

A method for predicting the server system resource demand in cloud computing is proposed, which builds a forecast using several prediction models such as ARIMA, ARIMAX, GMDH. When constructing a forecast, the method chooses the model that gave the most accurate forecast in the previous step.
Ключові слова: центр обробки даних, хмарні обчислення, прогнозування потреби ресурсів, моделі прогнозування, ARIMA, ARIMAX, GMDH.

1. Вступ

Створення центрів обробки даних (ЦОД) з десятками та сотнями тисяч комп'ютерних компонентів [1] актуалізувало проблему розробки ефективних методів управління їх ресурсами. При цьому в якості критерії ефективності зазвичай розглядається вимога щодо мінімізації використовуваних ресурсів ЦОД при задоволенні вимог до виконуваного в загальному випадку потоку програм. Для ЦОД загального призначення такими вимогами є досягнення максимальної реальної продуктивності (середня кількість програм, що виконуються за одиницю часу). Для кластерів і багатоядерних комп'ютерів критерії ефективності - це досягнення заданого (мінімального в теоретичному плані) часу виконання паралельної програми при умови використання мінімального кількості компонентів ЦОД.

Останнім часом завдяки швидкому розвитку обчислювальної техніки, технологій зберігання, зв'язку, хмарних обчислень, яка широко розглядається як третя технологічна революція після персонального комп'ютера та інтернету, і буде ключовою технологією ведення промислової революції протягом наступних 20 років. Хмарні обчислення були запропоновані як нова модель комерційної мережі для обчислення ресурсів [1,2].

Незважаючи на те, що хмарні обчислення останніми роками почали розвиватися значно швидше, проте вони стикаються з багатьма ключовими технічними проблемами, перше - це питання прогнозування потреби ресурсів серверних систем. Друге - це питання управління ресурсами, зокрема попит на правильне, швидке та автоматизоване балансування навантаження та ефективного розподілі ресурсів. Контроль та прогнозування ресурсів у середовищі хмарних обчислень є важливою частиною управління ресурсами, також є основою для інших методів управління ресурсами, її роль в основному[3]

Прогнозування потреби ресурсів в середовищі хмарних обчислень є важливою частиною управління ресурсами. Його роль в основному виявляється в наступних аспектах, перше - прогнозування потреби ресурсів, для виконання балансування навантаження в хмарних обчисленнях. Друге – за допомогою моніторингу та прогнозування можна розраховувати використання ресурсів різних клієнтів. Тому вивчення моніторингу ресурсів та прогнозування в середовищі хмарних обчислень має велике значення для вирішення проблем управління хмарними обчисленнями та сприяння розвитку хмарних обчислень.

Необхідні гнучкі рішення, які ґрунтуються на прогнозуванні потреби ресурсів та полягають у правильному розподілі навантаження і ефективному управлінні ресурсами. Для систематичного

прийняття правильних рішень необхідні інструментарій та комплекс методик і алгоритмів для вирішення задач управління IT-інфраструктурою. Їх створення становить важливу науково-практичну проблему, розв'язання якої вимагає розуміння процесів, які відбуваються в хостингових компаніях, функціонування IT-інфраструктури, чіткої постановки конкретних завдань дослідження, розробки математичних моделей, моделей ЦОД і відповідних методів вирішення задачі та реалізації згаданих вище методик і алгоритмів.[4]

Технології віртуалізації призначені для того, щоб користувачі могли абстрагуватися від особливостей окремих груп ресурсів, об'єднати їх в апаратно- програмні комплекси потрібної конфігурації і спростити управління цими групами ресурсів. Розрізняють віртуалізацію платформ, яка стосується процесу створення віртуальних машин (ВМ), і віртуалізацію ресурсів, яка переносить підходи до створення ВМ на усі види ресурсів (обладнання IT-інфраструктури, простори імен, мережі і т.п.). Загалом ВМ – це програмні абстракції, що запускаються на платформі реальних апаратно-програмних систем і емулюють роботу реального чи фіктивного апаратного забезпечення, використовуючи реальні ресурси хостової машини (ВМ – гостьова машина). Є декілька видів віртуалізації, які відрізняються повнотою симуляції апаратного забезпечення: віртуалізація (емуляція), яка розгалужується на повну, апаратну та часткову; паравіртуалізація; контейнерна віртуалізація (віртуалізація рівня ОС). [6]

Віртуалізація забезпечує зниження операційних витрат на підтримку парку серверів, розташування декількох серверних застосувань на одному фізичному сервері, підвищення доступності IT-сервісів і інші переваги [5].

2. Проблематика

Одним з самих складних технологічних питань в побудові серверних систем є питання прогнозування потреби ресурсів серверної системи, а відповідно і балансування. Через різні потреби користувачів віртуальних машин, різне навантаження на віртуальні машини, а також неоднорідності характеристик серверів стає дуже тяжко передбачити скільки ресурсів потрібно для кожної серверна. Ідеальною була б ситуація, коли всі ВМ однакові і створюють однакове навантаження, а всі обслуговуючі сервера мають однакові потужності. Однак, в ВМ постійно змінюється навантаження, а сервера, запущені в різний час, будуть мати різні обчислювальні потужності. Тому, без системи прогнозування навантаження не обійтися. Необхідно, щоб дана система прогнозування враховувала обчислювальні потужності кожного робочого вузла, а також поточне навантаження – створюване ВМ.

3. Алгоритм прогнозування на основі моделей аналізу часових рядів

Задача прогнозування часових рядів актуальна і вирішується на підставі моделі прогнозування. Одним з найбільш використовуваних класів моделей прогнозування є клас авторегресійних моделей – ARIMA, ARIMAX. А також метод GMDH. GMDH - застосовується у самих різноманітних областях для аналізу даних та знаходження знань, прогнозування і моделювання систем, оптимізації і розпізнавання образів.

Метою запропонованого алгоритму є поліпшення методів прогнозування потреби ресурсів серверної системи в умовах хмарних обчислень.

Під час роботи віртуальних машин, на них постійно збільшується та зменшується навантаження, відповідно це навантаження змінюється і на серверній системі. Щоб запобігти перевантаження або недовантаження серверної системи потрібно постійно мати актуальний та точний прогноз, що буде відбуватись в наступний період часу. Адже коли система недовантажена, йде простій ресурсів, які можна було б використати для інших ВМ. А коли перевантажена відбувається затримка доступу до сервіс, тобто порушення SLA. SLA виконується для кожного клієнта, коли вся продуктивність, яку потребують застосунки всередині ВМ, забезпечується в будь-який час.

Алгоритм прогнозування на основі моделей аналізу часових рядів наведено на рисунку 1.

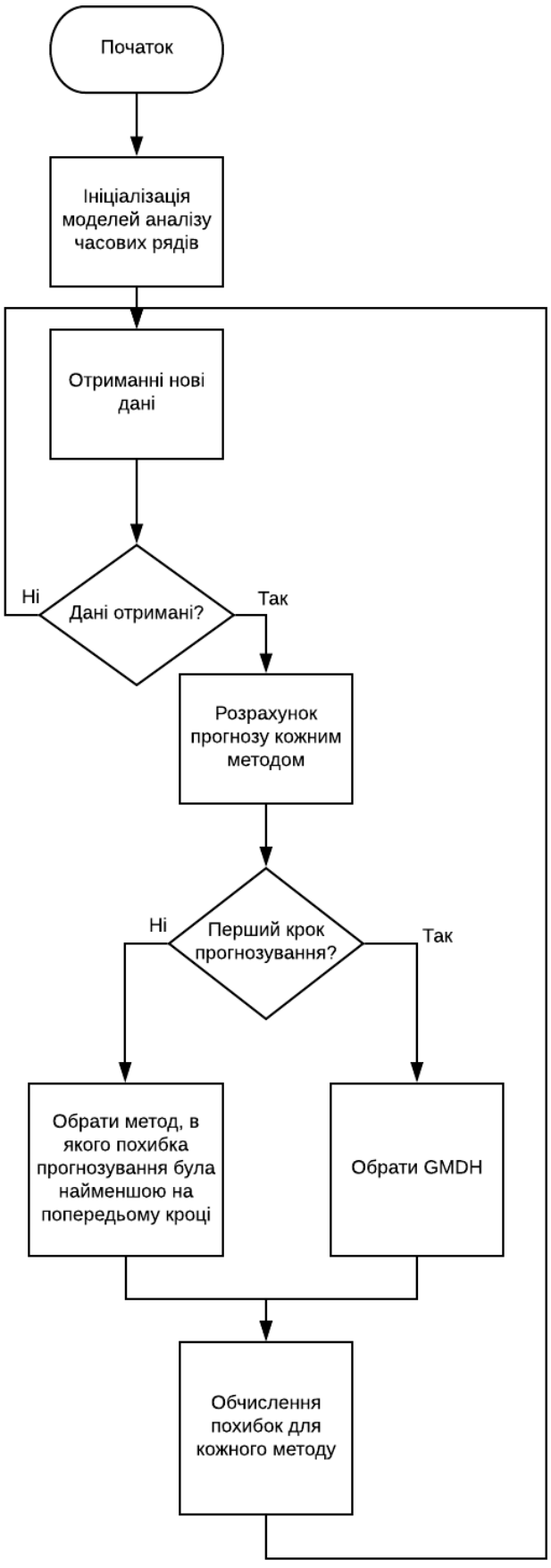


Рис. 1. Алгоритм прогнозування на основі моделей аналізу часових рядів

На вхід алгоритм отримує в режимі онлайн дані про завантаженість системи. Після кожного кроку прогнозування перевіряються похибки середня абсолютная помилка (англ. mean absolute percentage error, MAPE):

$$MAPE = \frac{1}{n}\sum_{i=1}^{n}\frac{|Y_i - \widehat{Y_{ij}}|}{Y_i} \qquad (1)$$

де n – кількість періодів;

Y_i – фактичне навантаження;

$\widehat{Y_{ij}}$ – прогнозоване навантаження j-го методу;

та середньо квадратична помилка (англ. root mean squared error, RMSE):

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(Y_i - \widehat{Y_i})^2}$$

4. Висновки

Проаналізовано можливість застосування методу прогнозування потреби ресурсів серверної системи в умовах хмарних обчислень. Запропоновано метод прогнозування потреби ресурсів серверної системи в умовах хмарних обчислень, який на кожному кроці прогнозування, перевіряє яка з моделей ARIMA, ARIMAX, GMDH дали найкращий результат. Після кожного кроку прогнозування, кожним алгоритмом перевіряється його точність. Перевагою методу прогнозування є здатність прогнозувати в онлайн режимі та додавати нові моделі прогнозування.

Програмно реалізований алгоритм прогнозування потреби ресурсів серверних систем

Список літератури

1. Теленик С.Ф. Управління навантаженням і ресурсами центрів оброблення даних при виділених серверах [Текст]: /С.Ф. Теленик, О.І. Ролік, М.М. Букасов, Р.В. Римар, К. О. Ролік.– Автоматика. Автоматизація. Електротехнічні комплексі та системі. – 2009. – №2 (24). – С.122 – 136.
2. Гузій М. М., Станіславова О. В., Кадет М. В. Аналіз технологій моніторингу комп'ютерних мереж [Текст]: /М. М. Гузій, О. В. Станіславова , М. В. Кадет.– В надзаг.: Наукоємні технології №1. - С.46 — 50.
3. Workload Prediction Using ARIMA Model and Its / R.Calheiros, E. Masoumi, R. Ranjan, R. Buyya. // IEEE Transactions on Cloud Computing. – 2015. – №3. – С. 449 – 458.
4. Workload Prediction for Cloud Computing Elasticity Mechanism / W. Dongxia Yazhou Hu, Bo Deng, Fuyang Peng // Cloud Computing and Big Data Analysis (ICCCBDA), IEEE. – 2016
5. Architecture and conceptual bases of cloud IT infrastructure management / Telenyk, S., Zharikov, E., Rolik, O. // Advances in Intelligent Systems and Computing. – 2017
6. E. Zharikov, O. Rolik and S. Telenyk, "An integrated approach to cloud data center resource management," 2017 4th International Scientific-Practical Conference Problems of Infocommunications. Science and Technology (PIC S&T), Kharkov, 2017, pp. 211-218.

УДК 51-37

ЯРУШЕВСЬКИЙ О.О.,
ЗАДІРАКА В.К.

ВИКОРИСТАННЯ АЛГОРИТМУ ШВИДКОГО ПЕРЕТВОРЕННЯ ФУР'Є ДЛЯ МНОЖЕННЯ ЦІЛИХ ЧИСЕЛ

Захист інформації значною мірою базується на використанні криптографічних методів, пов'язаних з шифруванням даних. У зв'язку з цим удосконалення існуючих методів шифрування та дешифрування є актуальною. Одним з підходів вирішення даної проблеми є застосування швидких алгоритмів обчислення криптопримітивів. Оскільки основними операціями для шифрування та дешифрування даних є операції множення та піднесення до степення. В даній статті розглянуто алгоритм швидкого множення цілих чисел з використанням швидкого перетворення Фур'є.

Information security is based on the use of cryptographic techniques related to encryption. In this regard, the improvement of existing methods of encryption and decryption is important. One of the approaches to solve this problem is the use of fast algorithms for calculating crypto-primitives. Since the basic operations for encryption and decryption of data is a multiplication operation and exponentiation. In this article, we consider the algorithm of fast multiplication of integers using a Fast Fourier Transform.

1. Вступ

У сучасному інформаційному суспільстві велику загрозу конфіденційності та цілісності інформації представляє кіберзлочинність. Зростання кількості кібератак та доступність програмно-технічних засобів для їх реалізації зумовлює необхідність розробки сучасних засобів інформаційної безпеки громадян та держави в цілому.

Отже, актуальною задачею є створення та вдосконалення систем захисту інформації, зокрема алгоритмів криптографічного захисту, що в більшості випадків є базовим ядром таких систем. На даний час основним питанням, що підлягає вирішенню, є збільшення об'ємів інформації, що може оброблятися функціями криптографічного перетворення. Одним з підходів вирішення даної проблеми є застосування швидких алгоритмів обчислення криптопримітивів.

В основу покладені алгоритми виконання різних операцій над багаторозрядними числами. Для простоти викладення припустимо, що ми маємо справу з цілими числами. Розглянуті алгоритми можна легко розповсюдити на випадок чисел з плаваючою комою тощо.

Під n розрядним цілим числом будемо розуміти довільне ціле число, менше за b^n , де b — основа прийнятої позиційної системи, в якій ми представляємо числа; такі числа в цій системі записуються з використанням не більше ніж n розрядів.

Ці числа можна розглядати як числа, записані в системі числення за основою b, де b — розмір машинного слова. Наприклад, ціле число, яке займає 10 машинних слів в пам'яті ЕОМ, розмір слова якої дорівнює $b = 10^{10}$, має 100 десяткових (цифр; але ми будемо розглядати його як десятирозрядне число за основою 10^{10}. Це робиться шляхом групування бітів.

Далі багаторозрядні цілі числа будемо називати ще багатослівними або s слівними, оскільки кожне з них можна представити у вигляді масиву 16-бітових, 32-бітових або 64- бітових слів.

На теперішній час досліджені алгоритми для наступних операцій:

– додавання або віднімання k-розрядних цілих чисел, з отриманням kрозрядної відповіді та цифри переносу;

– множення k-розрядного цілого числа на m-розрядне ціле число, з отриманням (m+k)-розрядної відповіді;

– піднесення k-розрядного числа до m-розрядного степеня;

– ділення (m+k)-розрядного цілого числа на m-розрядне ціле число з отриманням (m+1)-розрядної частки та k-розрядного лишку;

– виконання модулярних операцій.

Основна направленість дослідження - побудова ефективних за часом реалізації алгоритмів багаторозрядної арифметики. Як резерв оптимізації обчислень використовуються не тільки алгоритмічні новинки і різні моделі обчислень, але й нові архітектурні рішення, що дозволить побудувати ефективні за швидкодією алгоритми розв'язання задач інформаційної безпеки, а саме: підвищення продуктивності систем двоключової криптографії.

2. Представлення цілого числа у вигляді поліному

Існує два способи представлення поліномів: з допомогою коефіцієнтів та представлення ключ-значення.

Розглянемо перший підхід, нехай маємо *s*-слівне ціле числа a, тоді поліном буде мати вигляд:

$A(x) = a_0 + a_1x + a_2x^2 + \ldots + a_sx^s$

Наприклад, представимо число 348 у вигляді поліному з коефіцієнтами:

$A(x) = 8 + 4x + 3x^2$, де *x=10*

Інший підхід представлення поліному є ключ-значення. Маємо набір значень $\{(x_0, y_0), (x_1, y_1), \ldots, (x_n, y_n)\}$ причому виконуються наступні умови:

для всіх $i \neq j, x_i \neq x_j$. Тобто, всі точки унікальні;

для всіх *k*, $y_k = A(x_k)$.

Наприклад,

Маємо такі поліноми: $A(x) = x^3 - 2x + 1$, $B(x) = x^3 + x^2 + 1$, візьмемо 7 коефіцієнтів $x_k = \{-3, -2, -1, 0, 1, 2, 3\}$.

Далі представимо поліноми A та B у вигляді ключ-значення:

A: (-3, -20), (-2, -3), (-1, 2), (0, 1), (1, 0), (2, 5), (3, 22)

B: (-3, -17), (-2, -3), (-1, 3), (0, 1), (1, 3), (2, 13), (3, 37)

Якщо виконувати множення двох поліномів, що представленні як ключ-значення, то множення буде по точкове і час такого множення буде складати *O(n)*.

Повернемося до попереднього прикладу і виконаємо множення $C = A \cdot B$:

C: (-3, 340), (-2, 9), (-1, 2), (0, 1), (1, 0), (2, 65), (3, 814)

Приведення поліному до стандартного вигляду з коефіцієнтами називається інтерполяцією.

Якщо представити A та B, як вектори, тоді вектор C буде називатися математичною згорткою A та B (представляється, як $A \otimes B$).

3. Постановка задачі

Маємо два поліноми: $A(x) = a_0 + a_1x + a_2x^2 + \ldots + a_nx^n$ та $B(x) = b_0 + b_1x + b_2x^2 + \ldots + b_nx^n$, потрібно обчислити $C(x) = A(x) \cdot B(x)$, за час менший $O(n^2)$, що є часом прямолінійного множення, як показано в прикладі:

$$
\begin{array}{rrrrrrr}
 & & & 2x^2 & + 0x & + 5 & \\
 & & & 1x^2 & + 2x & + 3 & \\
 & & & - & - & - & \\
 & & & 6 & 0 & 15 & \\
 & & 4 & 0 & 10 & & \\
 & 2 & 0 & 5 & & & \\
\multicolumn{7}{l}{- - - - - - - - - - - - - -} \\
\multicolumn{7}{l}{2x^4 + 4x^3 + 11x^2 + 10x + 15}
\end{array}
$$

Більш загально, потрібно обчислити коефіцієнти c_i, коли $C(x) = A(x) \cdot B(x)$ та $C(x) = c_0 + c_1 + c_2x^2 + \ldots + c_{2n}x^{2n}$, тоді $c_i = a_0b_i + a_1b_{i-1} + \ldots + a_ib_0$.

4. Перетворення Фур'є та теорема про згортку

Перетворення Фур'є – інтегральне перетворення однієї комплексної

функції дійсної змінної на іншу, що дозволяє представити практично будь-

комбінацій таких тригонометричних функцій, як синус та косинус (перетворення Фур'є виходить за рамки даної статті).

Теорема про згортку. Нехай маємо функції f та g і їх згортку $\{f \otimes g\}$, позначимо перетворення Фур'є оператором $'\zeta'$. Тоді $\zeta(f)$ та $\zeta(g)$ є перетворенням Фур'є функцій f та g.

яку функцію у вигляді набору

Відповідно до теореми про згортку маємо:

$$\zeta(f \otimes g) = \zeta(f) \cdot \zeta(g)$$

де $\zeta(f) \cdot \zeta(g)$ - є по точковим множенням, що було розглянуто раніше.

5. Алгоритм множення

Загальна схема алгоритму показана на рис. 1.

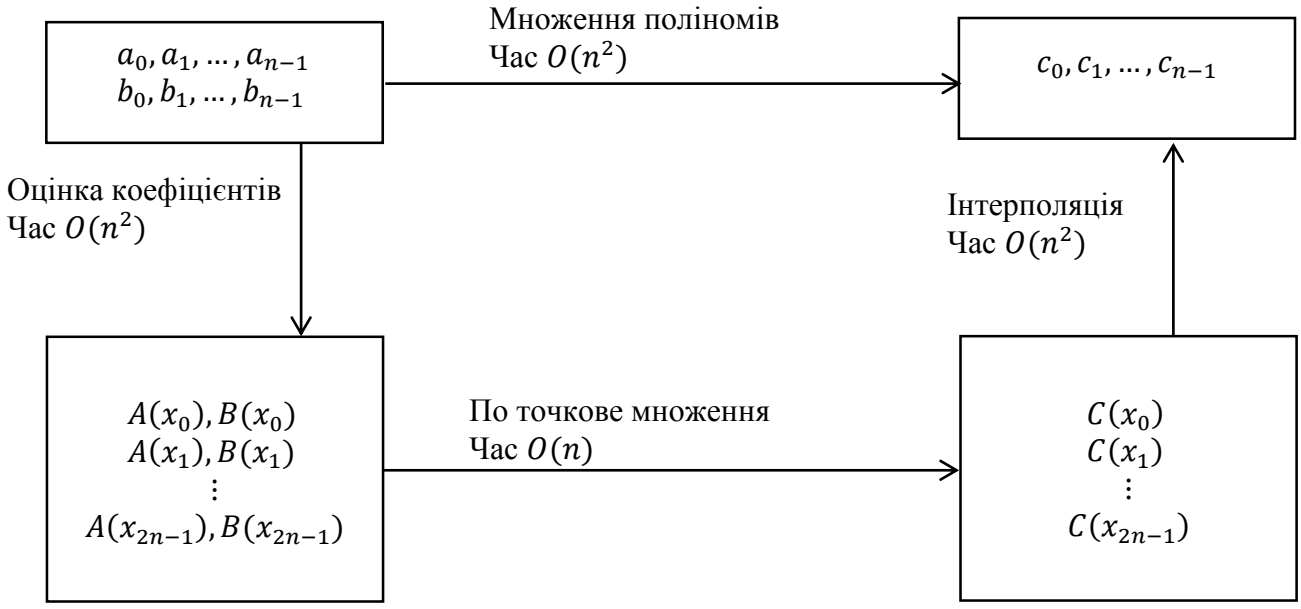


Рис 1. Схема алгоритму

Як бачимо складність алгоритму залежить саме від переходу від одного представлення поліному до іншого, в нашому випадку від представлення з коефіцієнтами до ключ-значення на етапі оцінки коефіцієнтів та в зворотньому на етапі інтерполяції. Тому загальна ефективність алгоритму буде залежати від ефективності конвертації між двома представленнями.

Тому для переходу від одного представлення до іншого використаємо алгоритм швидкого перетворення Фур'є, як вже відомо складність $O\big(n \cdot log(n)\big)$.

Отже, загальний алгоритм буде мати вигляд:

1. Оцінюємо значення поліномів *A* та *B* в *2n* точках, поліноми *A* та *B* повинні бути заповненні нулями до ступеня рівного *2n*, використовуючи алгоритм швидкого перетворення Фур'є. Складність $O\big(n \cdot log(n)\big)$.
2. Знаходимо значення поліному *C* в *2n* точках, виконуючи по точкове множення *A* на *B*. Складність $O(n)$.
3. Для інтерполяції застосуємо обернене перетворення Фур'є, використавши швидкий алгоритм перетворення Фур'є. Складність $O\big(n \cdot log(n)\big)$.

Використавши даний підхід, отримаємо прискорення по часу множення двох чисел від $O(n^2)$ до $O\big(n \cdot log(n)\big)$.

5. Висновок

В даній роботі було розглянуто спосіб прискорення алгоритму множення двох цілих чисел, використовуючи процедуру швидкого перетворення Фур'є. Завдяки такому підходу отримуємо резерв для оптимізації алгоритмів криптографії, що базуються в основному на операціях додавання, множення та піднесення до степеня.

Список літератури

1. Задірака В.К., Олексюк О.С. Комп'ютерна арифметика багаторозрядних чисел. //Київ - 2003
2. Карацуба А.А., Офман Ю.П. Умножение многорозрядных чисел на автоматах //ДАН СССР. — 1962. т.145. — С. 293-294.
3. Шенхаге А., Штрассен В. Быстрое умножение больших чисел // Кибернет. сб. — 1973. — вып. 10. — С. 87-98.
4. Cook S. A., Aanderaa S. O. On the minimum computation time of functions, Thesis, Harvard University, 1966. — P. 26-50.
5. J. W. Cooley and J. W. Tukey. An algorithm for the machine calculation of complex Fourier series. Mathematics of Computation, 19(90):297-301, 1965.
6. Кнут Д.Е. Искусство программирования для ЭВМ. Т.2.- М.: Издательский дом "Вильямс", 2001. - 828 с.

УДК 004.93

ГЛУХОВ В. О.
ХІМІЧ О. М.

ОГЛЯД АЛГОРИТМІВ НАВЧАННЯ ШТУЧНИХ НЕЙРОННИХ МЕРЕЖ

В даній статті розглянуто алгоритми оптимізації та засоби їх застосування для навчання штучних нейронних мереж з вчителем. Проаналізовано методи першого та другого порядків. Роглянуто їх переваги та недоліки. Розглянуто засоби апроксимації обрахування зворотнього Гессіану для методів оптимізаціїї другого порядку.

In this article described first and second orderd optimization algorithms and their application for the training of artificial neural networks. Examined their advantages and disadvantages. Overviewed simpler Hessian estimation.

1. Вступ

Машинне навчання є формою штучного інтелекту, що відповідає за можливість навчатися на основі даних. Такий процес зазвичай базується на основі двох компонент: параметризована та навчальна модель (наприклад згорткова нейронна мережа (CNN)) та відповідний алгоритм навчання. Структура моделі часто моделюється таким чином, щоби забезпечити нелінійну поведінку, що в свою чергу є важливим для вирішення задачі класифікації чи прогнозування.

Незалежно від того, наскільки якісна навчальна здатність у моделі, її ефективність навчання залежить від ефективності алгоритму навчання. Алгоритм навчання визначає, яким чином модель може оперувати ключовою інформацією в межах даних, та навчатися з отриманою статистики. Загалом є три основні види навчання: навчання без вчителя, напівавтоматичне навчання, та навчання з вчителем. Навчання без вчителя спрямоване на вивчення з нерозмічених даних, що часто ґрунтується на ймовірності виникнення шаблонів даних. Напівавтома-тичне навчання використовує невелику підмножину немаркованих даних, для забезпечення попереднього навчання моделі. Це надає можливість з'ясувати структуру прихованих шаблонів даних, перш ніж перейти до навчання з використанням розмічених даних. Навчання з вчителем відбувається завдяки аналізу пар вводу-виводу навчальних зразків [1]. Навчання відбувається завдяки аналізу міток для налаштування параметрів моделі. Ця робо-та присвячена розробці алгоритму навчання з вчителем для штучних нейронних мереж.

Більшість алгоритмів навчання спираються на ітеративні методи. З огляду на параметризовану модель, представлену як функцію, алгоритм навчання прагне знайти набір параметрів, що призводить цю функцію до оптимального рішення. У цьому контексті ця функція відноситься до штучних нейронних мереж (NN) як функція помилок (loss function), параметрами якої є вага та зміщення (bias), а рішенням є стан, де NN вважається добре навченою. Це робиться шляхом ітеративного знаходження відповідних зміщень для цих параметрів (вагів), доки не буде досягнуте бажане рішення:

$$W_{t+1} = W_t - \Delta W \qquad (1)$$

де W_t та W_{t+1} є вагами на t-ій та $(t+1)$-ій ітераціях відповідно, та ΔW що являє собою оптимальні значення зміщень, які слід вжити. Основна мета полягає в тому, щоб знайти найкраще ΔW так, щоб він досягав рішення в найкоротшій кількості ітерацій. У моделях NN, ΔW, як правило обчислюється шляхом знаходження градієнтів функції помилок відносно ваг, що може бути зроблено завдяки методу зворотного поширення помилки.

Градієнтний спуск - це алгоритм оптимізації першого порядку, який частіше всього використовується для навчання NN. Алгоритм простий, але дуже часто страждає повільною конвергенцією. На відміну від алгоритмів оптимізації першого порядку, алгоритми другого порядку для покращення швидкості

конвергенції використовують не лише значення градієнтів, але і кривізну . Як результат, ці алгоритми здатні покращити криву навчання, але за рахунок додаткових обчислень, які можуть бути надмірними для обчислення для глибинних нейронних мереж (таких як сучасні моделі CNN).

Додаткова проблема з більшістю цих складних алгоритмів навчання полягає в тому, що вони мають більше гіперпараметрів, ніж поширені алгоритми оптимізації першого порядку, що загрожує перенавчанню [2], в якому існують нескінченні способи налаштування алгоритму навчання, і це може призвести до вибору комбінації значень, які перевершують інші лише випадково.

У цій роботі ми розглядаємо алгоритм навчання другого порядку, що називається стохастичним діагональним алгоритмом Левенберга-Марквардта, та його модифікацію B-SDLM. SDLM був вперше запропановано Лекуном, як швидкий та ефективний метод навчання згортальних нейронних мереж (CNN) [3].

Основні відмінності модифікації:

1. B-SDLM заохочує швидку конвергенцію мережі, завдяки використанню як градієнту так і інформацію про кривизну функції, одночасно забезпечуючи її стабільність навчання за рахунок додаткового граничного стану;
2. B-SDLM полегшує проблему перенавчання пов’язану з налаштуванням гіперпараметрів, використовуючи лише один гіперпараметр, який слід налаштувати замість багатьох;
3. У порівнянні з звичайним алгоритмом SGD, обчислювальні накладні витрати в B-SDLM є незначними, оскільки вимагають лише незначну кількість навчальних зразків для апроксимації Гессіану.

2. Алгоритми навчання

Градієнтний спуск (GD) - це найпоширеніший спосіб оптимізації першого поряд-ку для навчання NN в режимі навчання з вчителем. Ідея градієнтного спуску полягає в тому, щоб прийняти зміщення, про-порційні негативному градієнту функції втрат $E(W_t)$ при поточній ітерації t для знаходження локального мінімуму (можливого рішення) функції:

$$W_{t+1} = W_t - \eta \frac{dE(W_t)}{dW_t} \quad (2)$$

де W_t та W_{t+1}є вагами на на t-ій та $(t+1)$-ій ітераціях відповідно, $E(W_t)$ - функція втрат, η - параметр, який відповідає за швидкість навчання, $\frac{dE(W_t)}{dW_t}$ - градієнт обчислений завдяки методу зворотного поширення помилки.

Звичайний GD працює в режимі пакетного навчання (Batch GD). BGD підсумовує значення градієнтів помилок для всіх виразів з навчальної вибірки, та потім усереднює їх. Він дуже легко розпаралелюється, проте його зближення до оптимального рішення дуже повільне. Під час усередення дєякі закономірності навчальних зразків можуть бути втрачені. При роботі з великими даними, BGD може бути незмінним для обчислень.

Стохастичний градієнтний спуск (SGD) виконує оновлення вагів, після обрахунків значень градієнтів помилок для кожного виразу з навчальної вибірки, або усереднює їх, якщо SGD працює в режимі mini-batch. Крива навчання буде не такою гладкою, на відміну від BGD, але як показує практика, це прискорює швидкість конвергенції завдяки більшим шансам уникнути локальних мінімумів [5]. Однак, для проблемних умов, SGD може страждати від повільної конвергенції біля місцевого мінімуму. Незважаючи на це, SGD є безумовно найпоширенішим методом навчання нейронних мереж з вчителем, і застосовується у поєднанні з алгоритмом зворотного поширення помилки (BP).

Значну роль у швидкості навчання NN грає вибір значення параметру η (learning rate). Параметр прискорення обумовлює в основному значення зміщень для досягнення оптимального рішення. Проте фіксоване значення цього параметру може не забезпечувати досягнення вирішення поставленої проблеми для усії прикладів, оскільки його значення залежить від стану помилки елементу та значень вагів на поточній ітерації. Тому для вирішення цієї проблеми прийнято

використовувати адаптивні засоби знаходження прискорення η. Їх можна поділити на два типи: глобальні та локальні адаптивні алгоритми.

2.1 Глобальні адаптивні алгоритми

Глобальні адаптивні алгоритми намагаються використовувати стан навчання для корегування глобальної швидкості навчання. Інший підхід полягає в тому, щоб змінити швидкість навчання з часом на основі фіксованого графіка. Швидкість навчання також може бути зменшена після того, як навчальний процес досягне плато, але це потребує додаткові гіперпараметри.

Для кожної ваги може знадобитися оптимальне значення зміщення, які можна знайти завдяки наступним виразам:

Power scheduling: $$\eta_t = \frac{\eta_o}{\left(1+\frac{t}{M}\right)^c} \quad (3)$$

Exponential scheduling $$\eta_t = \eta_0 \times 10^{-t/M} \quad (4)$$

де η_0 початкове значення прискорення, η_t- прискорення на t -ій ітерації. M розмір вибірки, яка використовується під час навчання на t -ій ітерації (зазвичай константе значення), c відповідає за контроль величини.

Інші алгоритми використовують різницю загальних помилок між поточним і попередніми ітераціями тренувань для налаштування швидкості навчання. Однак це може застосовуватися лише в режимі пакетного навчання та вимагає двох додаткових гіперпараметрів.

2.2 Локальні адаптивні алгоритми

Локальні адаптивні алгоритми, зазвичай використовують інформацію про вагу (наприклад часткові похідні) для виконання налаштування параметрів. Ця ідея дозволяє кожну вагу налаштувати індивідуально, відповідно до її поточної ефективності. Адаптивний субградієнтний метод (AdaGrad) налаштовує індивідуальне значення швидкості навчання η базуючись на значеннях частинних похідних від функції помилки E відносно вагів $W_{ji}^{(l)}$. Вище значення швидкості навчання накладається на ваги з меньшими частковими похідними і навпаки. В AdaGrad, швидкість навчання η обраховується наступним чином:

$$\left(\eta_{ji}^{(l)}\right)_t = \frac{\eta}{\sqrt{\sum \frac{dE}{dW_{ji}^{(l)}}}} \quad (5)$$

де $W_{ji}^{(l)}$- це ваги між між j-им нейроном в l-ому шару та i-им нейроном в $(l-1)$ шарі. $\left(\eta_{ji}^{(l)}\right)_t$ - це швидкість навчання для одної ваги $W_{ji}^{(l)}$ на t-ій ітерації.Знаменник являє собою градієнтне накопичення, яке забезпечує ефект відпалу. Втім він має недолік, який в кінцевому рахунку зникає після багатьох тренувальних ітерацій. Ретельний підбір глобального параметру швидкості навчання має вирішальне значення, оскільки AdaGrad може бути чутливим до початкових значень вагів та їх відповідних градієнтів.

2.3 Алгоритми оптимізації другого порядку

Алгоритми оптимізації другого порядку зазвичай враховують як значення градієнтів так і інформацію про кривизну функції. Зазвичай застосовують методи Ньютона або квазі-Ньютона для обчислення унікальних значення прискорень для оновлення кожних вагів. Отже, вони є також локальними адаптивними алгоритмами.

2.3.1 Метод Ньютона-Рафсона

Метод Ньютона спрямований на пошук напрямку таким чином, щоб він завжди вказував на оптимальне рішення. Глобальний мінімум функції можливий лише тоді, коли $\frac{dE(W\ )}{dW} = 0$, таким чином оптимальні значень вагів можливо обрахувати наступним чином:

$$W_{opt} = W_t - \left(\frac{d^2E(W_t)}{dW^2}\right)^{-1} \frac{dE(W_t)}{dW} \quad (6)$$

Звідси випливає:

$$\eta_{opt} = \left(\frac{d^2E(W_t)}{dW^2}\right)^{-1} \quad (7)$$

Отже для обрахунку оптимального значення локального прискорення необхідно обрахувати зворотній Гессіан.

Метод Ньютону гарантує

конвергенцію для опуклих функцій, але більшість функцій за своєю природою не є такими (в тому числі функції помилок в NN). Обрахування самого Гессіану і його зворотної матриці є дуже ресурсо-затратними $O(n^3)$ і на практиці не придатний для вирішення задач оптимізації оптимізації сучасних топологій мереж.

2.3.2. Апроксимація Гессіану

Існує ряд методів апроксимації Гессіану, призначених зменшити складність розрахунків. Розглянемо апроксимацію за допомогою методу Левенберга-Марквардта. Він базується на апроксимації GN (Gauss-Newton), але використовує додатковий параметр регуляризації μ, щоб уникнути спалаху (blowing up) у тому випадку, коли частинні похідні другого порядку занадто малі.

$$\Delta W = \sum_M \left(\frac{\partial f(W,X_m)^T}{\partial W}\frac{\partial f(W,X_m)}{\partial W}\right) + \mu I^{-1}\frac{dE(W)}{dW}$$

Апроксимація Гессіану здійснюється шляхом скидання недіагональних значень. Отже діагональна матриця Гессіану розраховується шляхом зворотного поширення помилки другого порядку:

$$\Delta W = \frac{\frac{\partial E}{\partial W_{ji}^{(l)}}}{\left|\frac{\partial^2 E}{\partial W_{ji}^{(l)2}}\right| + \mu} \quad (8)$$

3. Висновок

Таким чином у статті було розглянуто використання методів оптимізації для навчання штучних нейронних мереж. Для цього було проведено розбір та аналіз існуючих методів. Були з'ясовані переваги та недоліки методів другого порядку над методами першого порядку. Було виявлено велику складність обрахунків зворотнього Гессіану та розглянуто можливі варіанти апроксимації.

Використання таких алгоритмів як SDLM та його модифікації дозволить швидше навчати сучасні нейронні мережі з мінімальними додатковими затратами.

Список літератури

1. C. Hong, J. Yu, J. Wan, D. Tao, M. Wang, Multimodal deep autoencoder for human pose recovery, IEEE Trans. Image Process. 24 (12) (2015)

2. J. Jäderbo, Tuning of learning algorithms for use in automated product recommendations, Student Paper, 2014.

3. LeCun, Y. Gradient-Based Learning Applied to Document Recognition / [Y. LeCun, L. Bottou, Y. Bengio and P. Haffne] – Proc. IEEE 1998. – (Препринт / IEEE; 1998).

4. C. Igel, M. Hsken, Improving the Rprop learning algorithm, in: Proceedings of the Second International Symposium on Neural Computation (Nc), 2000, pp. 115–121.

5. LeCun, Y. Efficient BackProp / Y. LeCun, L. Bottou, G. Orr and K. Muller – Neural Networks: Tricks of the trade, Springer, 1998.

6. Khalil-Hani M. A-SDLM: An Asynchronous Stochastic Learning Algorithm For Fast Distributed Learning / [M. Khalil-Hani, S.S. Liew] – AusPDC 2015. – (Препринт / AusPDC; 2015).

7. Khalil-Hani M. An optimized second order stochastic learning algorithm for neural network training / [M. Khalil-Hani, S.S. Liew, R. Bakhteri] – Neurocomputing 2016. – (Препринт / Neurocomputing; 2016).

УДК 519.68; 681.513.7; 612.8.001.57; 007.51/.52

ДІДКІВСЬКА В.А.,
ГАВРИЛЕНКО О.В.

ПІДХІД ДО ПРОГНОЗУВАННЯ РЕЗУЛЬТАТІВ СПОРТИВНИХ ЗМАГАНЬ

У даній статті розглянуто підхід до прогнозування результатів спортивних змагань за допомогою марківських процесів та нейронних мереж. Демонструються результати прогнозування та опис підходу. На прикладі прогнозування результатів тенісного матчу наведено кількісні розрахунки.

This article describes the approach on predicting the results of sports competitions using Markov processes and neural networks. The results of prediction and description of the approach are shown. The experiment shows the prediction results for tennis match, quantitative calculations are providen.

1. Вступ

У даній статті розглянуто підхід до прогнозування результатів спортивних змагань за допомогою марківських процесів та нейронних мереж. Створений підхід може бути використаний для аналізування різноманітних видів спорту. Використання може бути можливим завдяки впровадженню в систему правил та значень, які відповідають матчу виду спорту, який має бути проаналізований. Як приклад, у даній роботі було взято за основу великий теніс: його правила, виключні ситуації, побудову гри. На прикладі тенісної гри наведено результати прогнозування та їх порівняння.

2. Актуальність створення підходу до прогнозування та загальна методологія

Прогнозування результатів спортивних змагань – область досліджень, яка стрімко розвивається і річ не тільки в ставках на спорт: застосування математики та результатів прогнозів допомагає досягати спортсменам кращих результатів. За допомогою прогнозування можливо визначити “слабкі” місця спортсмена у кількісних значеннях, мету, якої варто досягти до наступних змагань та запропонувати шляхи або засоби, за допомогою яких це досягнення може стати можливим.

Система очок в тенісі має ієрархічну структуру: матч складається з сетів, які складаються з геймів, які складаються з окремих очок. У більшості сучасних підходів до прогнозування тенісу ця структура використовується для отримання ієрархічних виразів ймовірності перемоги гравці у матчі на основі марківських ланцюгів. Якщо вважати, що очки в тенісі розподіляються незалежно і однаково, для отримання виразу необхідно знати тільки ймовірність виграшу кожним гравцем очки при подачі.

Альтернативним підходом для стандартного методу марківських ланцюгів є машинне навчання. Параметри гравців матчу разом з результатом матчу можуть скласти навчальну вибірку. Алгоритм машинного навчання з учителем може використовувати цю вибірку для побудови функції передбачення результатів нових матчів.

Сучасні моделі прогнозування тенісу засновані на описаних ієрархічних стохастичних виразах. Кнottенбельт [1] уточнив моделі Барнета, використавши для обчислення ймовірності виграшу очка при подачі тільки матчі зі спільними суперниками гравців, замість всіх минулих суперників. Цей підхід дозволяє знизити похибка, що виникає через те, що гравці в минулому зустрічалися з суперниками різного рівня. Мадурська [2] далі розширила модель загального суперника Кнottенбельта, використавши різні ймовірності виграшу очка при подачі для різних сетів. Таким чином, автор відмовилася від допущення незалежного однакового розподілу очок і її модель відображає накопичення фізичної втоми у гравця впродовж матчу.

Підхід до прогнозування тенісного матчу, що буде описаний далі, поєднує в собі як марківські процеси, так і нейронні

мережі. Він базується на існуючих методах прогнозування та їх покращенні для підвищення точності прогнозу. Таким чином, можна виявити "слабкі" місця гравця та запропонувати їх поліпшити задля підвищення результату. Дана методологія може бути описана послідовністю наступних кроків:

1. Збір даних – формування вибірки даних, базуючись на загальнодоступній статистиці тенісних турнірів для подальшого коректного аналізу.
2. Прогнозування результатів за допомогою модифікованого та стандартного методів ланцюгів Маркова – обраний матч-поєдинок зі статистики аналізується та проводиться прогнозування, перевірка достовірності результатів – звірення з існуючими історичними даними статистики.
3. Прогнозування результатів за допомогою нейронної мережі - матч аналізується та проводиться повторне прогнозування, при цьому враховується більша кількість факторів, які можуть впливати на результат матчу (покриття корту, фізичний стан гравця тощо). Перевірка результатів – згідно з даними прогнозування методом ланцюгів Маркова, аналіз похибки прогнозу.
4. Надання рекомендацій на основі проведених прогнозів – формування вибірки факторів, зміна яких може позитивно вплинути на результат матчу.
5. Повторне прогнозування результатів за допомогою модифікованого та стандартного методів ланцюгів Маркова – проведення повторного прогнозування з метою перевірки коректності наданих рекомендацій щодо зміни кількісних значень факторів.
6. Повторне прогнозування результатів за допомогою нейронної мережі – виявлення нових можливих факторів, які можуть покращити результат матчу.

3. Прогнозування тенісного матчу за допомогою ланцюгів Маркова

Візьмемо тенісний матч проти одного суперника (сингл) - одиночний тенісний матч.

Припустимо, що ми розглядаємо матч, у якому приймають участь два гравці тенісного матчу, перший гравець - *A*, другий гравець - *B*. Припустимо, що p - ймовірність того, що гравець A виграє очко, якщо подаватиме. Відповідно, припустимо, що q - ймовірність того, що гравець B виграє очко, якщо подаватиме. Нехай ймовірність виграшу очка незалежно та однаково розподілена.

Така постановка дає нам можливість побудувати ланцюг Маркова, за допомогою якого буде описано ймовірність виграшу гравця в геймі.

У тенісі гейм — відрізок сету, в якому право подачі належить виключно одному із супротивників, який повинен чергувати її, подаючи в ліву й праву половину корту. Для перемоги у сеті потрібно виграти принаймні 6 геймів і принаймні на два більше, ніж супротивник. При рахунку 6:6 за геймами у більшості турнірів грається тай-брейк.

Вважатимемо рахунок простором станів, тоді переходи між станами – це ймовірності виграшу p першого гравця A або ймовірності виграшу q другого гравця B. Усі переходи, які означають очко, вигране гравцем A, мають ту ж ймовірність p, а всі переходи, які означають програне очко, мають ймовірність $1-p$. Візуалізацію ланцюга Маркова за умови, що подає перший гравець А, наведено на рис.1:

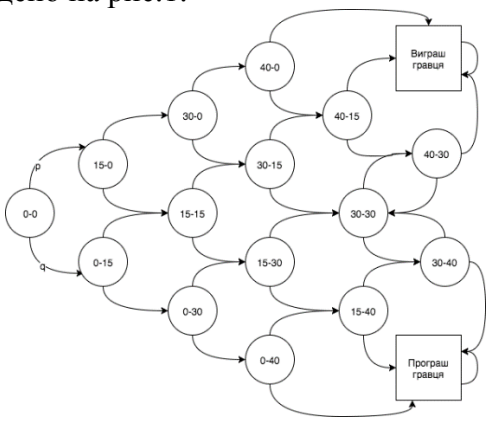


Рис. 1. Ланцюг Маркова за умови подачі першого гравця

Виходячи з ідеї моделювання тенісних матчів з ланцюгами Маркова, Барнетт і Кларк [3] та О'Малі [4] розробили ієрархічні вирази для ймовірності того, що конкретний гравець виграє весь тенісний матч. Барнетт і Кларк виражають ймовірність того, що гравець A виграв гру при своїй подачі $P_{\text{гейм}}$, використовуючи наступне рекурсивне визначення:

$$P_{\text{гейм}}(x,y) = p \cdot P_{\text{гейм}}(x+1,y) + (1-p)P_{\text{гейм}}(x,y+1)$$

Граничні значення записуються як:

$$P_{\text{гейм}}(x,y) = 1, \text{якщо } x = 4, x - y \geq 2$$

$$P_{\text{гейм}}(x,y) = 0, \text{якщо } y = 4, y - x \geq 2$$

$$P_{\text{гейм}}(x,y) = \frac{p^2}{p^2+(1-p)^2}, \text{якщо } x = 4, x - y \geq 2$$

У наведеному вище, p - ймовірність того, що гравець *A* виграє очко при подачі, а x та y представляють кількість очок, вигранних гравцями *A* та *B*, відповідно. Цей вираз явно відповідає ланцюжку Маркова.

Отже, дані ієрархічні вирази, отримані Барнеттом і Кларком, дозволяють визначити переможця. Залишається питання - як оцінити ці ймовірності виграшу очка при подачі для ще не зіграних матчів? Барнетт і Кларк дають ефективний метод для оцінки цих імовірностей з історичної статистики гравців:

$$f_i = a_i b_i + (1 - a_i)c_i$$

$$g_i = a_{av} d_i + (1 - a_{av})e_i \text{ , де}$$

f_i - відсоток очок, вигранних при подачі гравцем *i;*

g_i - відсоток очок, вигранних при прийомі м'яча гравцем *i;*

a_i - відсоток перших подач гравця *i;*

a_{av} - средний відсоток перших подач для всіх гравців*;*

b_i - відсоток виграшу при першій подачі гравця *i;*

c_i - відсоток виграшу при другій подачі гравця *i;*

d_i - відсоток виграшу при прийомі першої подачі гравцем *i;*

e_i - відсоток виграшу при прийомі другої подачі гравцем *i.*

Отже, для матчу між гравцями *A* і *B* ми можемо оцінити ймовірності виграшу очка при подачі гравцями *A* і *B* відповідно як f_{AB} і f_{BA}, використовуючи наступне рівняння:

$$f_{AB} = f_t + (f_i + f_{av}) - (g_i - g_{av}) \text{ , де}$$

f_t - відсоток очок, вигранних при подачі гравцем *i;*

f_{av} - відсоток очок, вигранних при прийомі м'яча гравцем *i;*

g_{av} - відсоток перших подач гравця *i.*

При моделюванні тенісного матчу також необхідно розглядати ситуацію тайбрейку.

Тайбрейк грається при рахунку 6:6 за геймами в сеті. При тайбреку кожен із гравців подає дві подачі в праву та ліву частини корту, після чого право подачі передається супротивнику. Зміна половинами корту відбувається після кожних шести подач. Ведеться підрахунок вигранних м'ячів, і для перемоги потрібно виграти 7 м'ячів, але принаймні на два більше, ніж у супротивника.

Згідно з попередніми міркуваннями про гейм тенісного матчу, рекурентні формули для тайбрейку гейму матимуть вигляд:

$$P_A^{pgT}(a,b) = p_A P_B^{pgT}(a+1,b) + q_A P_B^{pgT}(a,b+1)$$

$$p_A^{pgT}(a,b) = p_A P_A^{pgT}(a+1,b) + q_A P_A^{pgT}(a,b+1)$$

$$P_A^{pgT}(a,b) = 1, \text{якщо } a = 7, 0 \leq b \leq 5$$

$$P_A^{pgT}(a,b) = 0, \text{якщо } b = 7, 0 \leq a \leq 5$$

$$P_A^{pgT}(6{,}6) = p_A q_B / (p_A q_B + q_A p_B)$$

$$q_B = 1 - p_B$$

Рекурентні формули для тайбрейку сету матимуть вигляд:

$$P_A^{gsT}(c,d) = P_A^{pg}(0{,}0)P_B^{gsT}(c+1,d) + [1 - P_A^{pg}(0{,}0)]P_B^{gsT}(c,d+1)$$

$$P_A^{gsT}(c,d) = 1, \text{якщо } c = 6, 0 \leq d \leq 4 \text{ або } c = 7, d = 5$$

$$P_A^{gsT}(c,d) = 0, \text{якщо } d = 6, 0 \leq c \leq 4 \text{ або } c = 5, d = 7$$

$$P_A^{gsT}(6{,}6) = P_A^{pgT}(0{,}0)$$

Візуалізація марківської моделі для тайбрейку наведена на рис. 2:

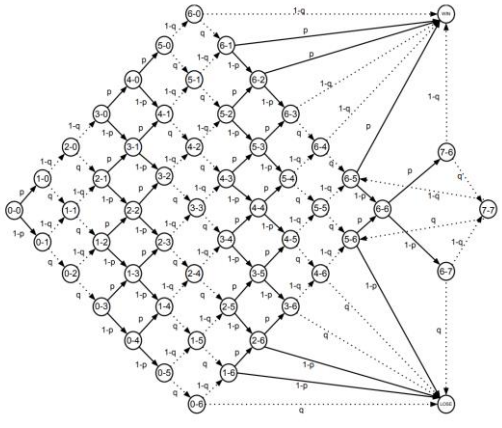


Рис. 2. Ланцюг Маркова при ситуації тайбрейку

Сет або партія — частина матчу, матч розбивається на кілька сетів і гра продовжується доти, доки один із супротивників не виграє певну їхню кількість (2-3, залежно від правил гри чи окремого турніру). Тенісний сет складається

з геймів і грається до перемоги одного із супротивників принаймні у 6 геймах. При цьому необхідно, щоб кількість виграних принаймні на два перевищувала кількість програних геймів. При рахунку 6:6 за геймами більшість сучасних тенісних турнірів передбачає тайбрейк. Візуалізація марківської моделі для сету наведена на рис. 3

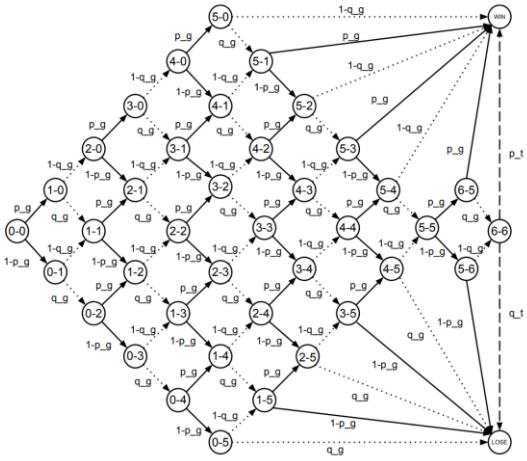


Рис. 3. Ланцюг Маркова при моделюванні сету

4. Модифікація методу ланцюга Маркова

Візьмемо за основу гейм і вважатимемо рахунок простором станів. Побудуємо модель для гравця *A* за умови, що не він подавав (прийняв) – оберемо деяке *s*, що означатиме ймовірність виграшу очка гравцем *A* за умови, що подає (приймає) не він, а суперник *B*. Ймовірність виграшу очка суперником B позначимо як l.

Побудуємо марківський процес із дискретним часом, кінцевим станом та матрицею переходів *M* і початковим станом I_0. Множина станів цього ланцюга являється поглинаючою (або замкненою), так як включає в себе 2 стани, з яких не може бути переходу в наступний стан – це стан виграшу гравця *A* та стан виграшу гравця *B*. Оскільки ми модифікуємо ситуацію гри гейму, то матриця *M* набуває вигляду:

$$M = \begin{bmatrix} m_{1\,1} & \cdots & m_{1\,17} \\ \vdots & \ddots & \vdots \\ m_{17\,1} & \cdots & m_{17\,17} \end{bmatrix}$$

Це зумовлено тим, що при моделюванні марківського ланцюга для гейму гри може бути всього 17 станів системи. Запишемо загальний вигляд блочної структури даної матриці:

$$M = \begin{bmatrix} E & 0 \\ R & W \end{bmatrix}$$

Тут E - одинична підматриця, порядок якої збігається з числом поглинаючих станів; W - квадратна підматриця ймовірностей переходів на множині неповоротних станів; R - прямокутна підматриця переходів з неповоротних станів в поглинаючі; 0 - нульова підматриця. Якщо число загальних станів – 17, з них 2 – поглинаючі стани, тоді підматриці мають наступний порядок:

$$M_{17*17},\ E_{15*15},\ W_{2*2}, R_{2*15}, 0_{15*2}$$

5. Нейронні мережі у прогнозуванні спортивних змагань

Прогнозування за допомогою нейронних мереж має безперечну перевагу над іншими способами прогнозування – це можливість враховувати різноманітні фактори, пов'язані з грою: стан поля і його покриття, виснаження, гравця тощо. Подібні данні можуть скоригувати прогноз та дати точніший результат. Сіпко[5] у своїй роботі подає метод прогнозування результатів тенісних матчів за допомогою нейронних мереж.

Для навчання потрібно мати набір промаркованих прикладів для навчання. В контексті прогнозування тенісу кожен навчальний приклад відповідає одному історичному тенісному матчу і складається з двох елементів:

1. Вектор вхідних функцій (X), що представляють характеристики гравців і матч
2. Цільове значення (y), що відповідає результатам матчу.

Цільове значення може набувати 1 або 0 (виграш або програш відповідно). Тренована модель може бути використана для прогнозування результатів майбутнього матчу.

Нейронна мережа прямого поширення (тобто мережа з механізмом прогнозування подій, feed-forward network) - це орієнтований ациклічний граф. Візуалізацію тришарової нейронної мережі наведено на рис.4

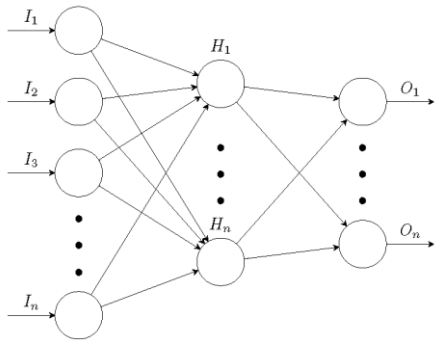


Рис. 4. Тришарова нейронна мережа

Кожному зв'язку в мережі присвоюються ваги. Нейрон використовує вхідний сигнал і його вагу для обчислення значення на виході. Типовим методом компонування мережі є нелінійна зважена сума:

$$f(x) = K(\sum_i w_i\, x_i)$$, де

w_i – вага вхідних даних x_i.

Нелінійна функція активації K дозволяє мережі обчислювати нетривіальні завдання, використовуючи малу кількість нейронів. Зазвичай для цієї мети використовуються сигмоїдальні функції, наприклад, логістична функція.

Тенісні матчі можна прогнозувати, подаючи на вхідний шар нейронів ознаки гравця і матчу, а потім проводячи значення через мережу. Якщо використовувати логістичну функцію активації, значення на виході мережі може являти собою ймовірність перемоги в матчі. Існує багато різних алгоритмів навчання, метою яких є оптимізація ваг мережі для отримання найкращих значень на виході для навчальної вибірки. Наприклад, алгоритм зворотного поширення використовує градієнтний спуск для зниження середньоквадратичної помилки між цільовими значеннями і значеннями на виході нейронної мережі.

6. Логістична регресія при нейромережевому підході

При нейромережевому підході прогнозування будується за допомогою логістичної регресії. Логістична регресія - це по суті алгоритм класифікації. Головними в алгоритмі є властивості логістичної функції. Логістична функція σ (t) визначається як:

$$\sigma(t) = 1/(1 + e^{-t})$$

Логістична функція відображає суттєві вхідні значення в діапазоні від -∞ до + ∞ і від 0 до 1, дозволяючи інтерпретувати виходи як ймовірності.

Модель логістичної регресії для - прогнозування матчів складається з вектора n ознак матчу $x = x_0, x_1, \dots, x_n$ і вектора $n + 1$ речових параметрів моделі $\beta = \beta_0, \beta_1, \dots, \beta_n$. Для прогнозування за допомогою моделі спочатку проектуємо точку в нашому n-розмірному просторі ознак на дійсне число:

$$z = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_n x_n$$

Тепер можна перетворити z в значення в прийнятному діапазоні ймовірностей (від 0 до 1) за допомогою логістичної функції, визначеної вище:

$$p = \sigma(z) = 1/(1 + e^{-z})$$

Навчання моделі складається з оптимізації параметрів β, так щоб модель давала найкраще відтворення результатів матчів для навчальної вибірки. Це здійснюється шляхом мінімізації функції логістичних втрат (рівняння нижче), яка дає міру похибки моделі при прогнозуванні результатів матчів, що використовувалися для навчання.

7. Експеримент

Було проаналізовано поєдинок між двома гравцями. Було взято статистичні дані для аналізу [6] та проаналізовано один із матчів для гравців А і В. Спершу було побудовано марківську модель та розраховано ймовірність виграшу матчу із урахуванням статистичних ймовірностей виграшу при подачі та виграшу при прийомі гравців. Проаналізувавши дані звичайним та модифікованим методом марківських ланцюгів, було виявлено ймовірності виграшу очка гравцем при подачі та при прийомі. Після цього, проаналізувавши дані за допомогою нейронної мережі, було виявлено два можливих показники, які варто підвищити для отримання кращого результату для гравця А: запропоновано тренувати другу подачу та перші прийоми. Дійсно, повторний аналіз показав, що результат може бути покращений завдяки підвищенню виявлених показників. Далі наведено таблиці статистичних даних та результати прогнозування.

Таблиця 1. Таблиця статистичних даних гравців

Гравець/ Характеристика	**Виграш при подачі**	**Виграш при прийомі**	**Ейси**	**Подвійні помилки**	**Очко при першій подачі**	**Очко при другій подачі**	**Очко при першому прийомі**	**Очко при другому прийомі**
Гравець А	0,77	0,41	3	2	0,67	0,32	0,23	0,62
Гравець В	0,23	0,59	20	8	0,77	0,38	0,33	0,68

Таблиця 2. Таблиця умовних імовірностей отримання очка при подачі

Гравець А	**Гравець В**					
		0	15	30	40	Гейм
	0	0.962890674	0.896661648	0.737136081	0.419136599	0
	15	0.98267337	0.944312142	0.832122939	0.544333246	0
	30	0.994131919	0.977823203	0.918086095	0.706926293	0
	40	0.999003354	0.995666754	0.981159802	0.918086095	
	Гейм	1	1	1		

Таблиця 3. Таблиця умовних імовірностей отримання очка при прийомі

Гравець А	**Гравець В**					
		0	15	30	40	Гейм
	0	0.285501053	0.173073318	0.081225575	0.022444053	0
	15	0.447287306	0.30524446	0.165813619	0.054741592	0
	30	0.651690425	0.505888841	0.325648973	0.133516079	0
	40	0.86150246	0.765258408	0.602132894	0.325648973	
	Гейм	1	1	1		

Таблиця 4. Таблиця порівняння показників гравця А

Гравець/ Характеристика	**Очко при другій подачі**	**Очко при першому прийомі**
Статистичні показники гравця А	0,32	0,23
Покращені дані гравця А	0,45	0,37

Таблиця 5. Таблиця порівняння результатів прогнозування за допомогою нейромережі для гравця А

Гравець/ Характеристика	**Виграш при подачі**	**Виграш при прийомі**
Ймовірність виграшу гравця А до покращення показників	91.32%	27.55%
Ймовірність виграшу гравця А після покращення показників	93.7%	29.23%

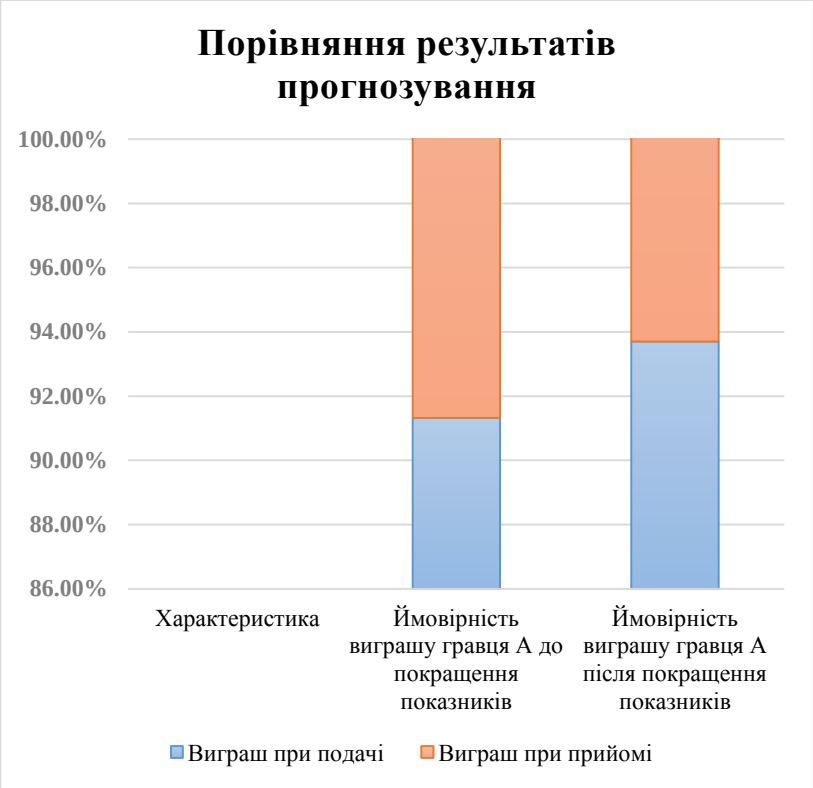


Рис. 5. Графік порівняння результатів прогнозування

8. Висновки

Було наведено методологію прогнозування результатів спортивних змагань на прикладі гри в теніс. За допомогою такого підходу до прогнозування стає можливим визначити "слабкі" сторони гравця та промоделювати ситуацію з покращеними показниками. Комбінування стохастичного методу прогнозування, а саме методу ланцюгів Маркова, та нейромережевого підходу дають коректні результати, які можуть бути використані при підготовці до турнірів гравцем. Перспективою розвитку такого підходу може бути зміна архітектури побудови нейронної мережі та зміна точки зору на сам ігровий процес з пошуком можливих нових алгоритмів прогнозування (врахування зонування тенісного корту, аналіз можливого формування пар гравців тощо).

9. Перелік літератури

1. W. J. Knottenbelt, D. Spanias, and A. M. Madurska. A common-opponent stochastic model for predicting the outcome of professional tennis matches. Computers and Mathematics with Applications, 64:3820–3827, 2012.
2. A. M. Madurska. A Set-By-Set Analysis Method for Predicting the Outcome of Professional Singles Tennis Matches. Technical report, Imperial College London, London, 2012.
3. T. Barnett and S. R. Clarke. Combining player statistics to predict outcomes of tennis matches. IMA Journal of Management Mathematics, 16:113–120, 2005.
4. J. A. O'Malley. Probability Formulas and Statistical Analysis in Tennis. Journal of Quantitative Analysis in Sports, 4(2), 2008.
5. M. Sipko. Machine Learning for the Prediction of Professional Tennis Matches. Technical report, Imperial College London, London, 2015.
6. ATP World Tour: Tennis Statistics. [Електронний ресурс] – Режим доступу: http://www.atpworldtour.com

УДК 519.854.2

ДУБОК К.В.,
ЖДАНОВА О.Г.,
СПЕРКАЧ М.О.

ЗАДАЧА СКЛАДАННЯ РОЗКЛАДУ ВИКОНАННЯ РОБІТ З ВІДНОШЕННЯМ ПЕРЕДУВАННЯ ПАРАЛЕЛЬНИМИ ПРИСТРОЯМИ ЗА КРИТЕРІЄМ МІНІМІЗАЦІЇ ЗАГАЛЬНОГО ЧАСУ ВИКОНАННЯ РОБІТ

Розглядається задача календарного планування виконання множини робіт паралельними пристроями різної продуктивності з метою побудови розкладу із мінімальним загальним часом виконання робіт. На роботи накладено відношення передування у вигляді ланцюгів. Розглянуті властивості досліджуваної задачі. Визначені достатні умови оптимальності розкладів. Розроблені алгоритми побудови початкового розкладу.

Scheduling problem of execution set of jobs on parallel machines with different speeds for constructing a schedule to minimize makespan. The jobs has a precedence relation in the form of chains. Propertiesed of the investigated problem are considered. Sufficient conditions for optimality of schedules are defined. Algorithms for constructing an initial decomposition are developed.

Ключові слова: календарне планування, розклад, робота, відношення передування, паралельні пристрої різної продуктивності, загальний час виконання робіт

1. Вступ

Успіх виробництва та його прибутковість на пряму залежить від процесу планування роботи на ньому. Ключовою складовою процесу планування є розробка ефективного плану виконання робіт виробничої діяльності. Для створення таких планів застосовують різні методи, зокрема методи теорії розкладів. Більшість задач теорії розкладів відносяться до класу NP-повних задач. На даний час існує багато підходів та методів для розв'язання задач теорії розкладу [1-5]. Робота науковців різних країн світу над даною проблемою свідчить про її актуальність та потребу в подальших дослідженнях.

2. Постановка задачі

Дано плоску конструкцію, на якій можна розмістити предмети, для їх подальшого переміщення вантажопідйомним пристроєм (далі палета). Палета складається з S рядів, кожен ряд має H комірок (рис. 1).

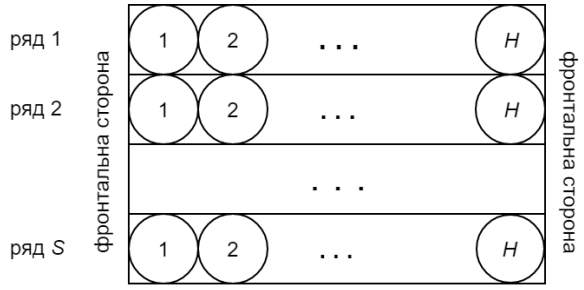


Рис. 1. Схема палети

Завантаження палети здійснюється тільки зі фронтальних сторін. У загальному випадку кожен ряд палети може завантажуватися з деякої точки, яка розділяє палету на дві частини: перша частина містить q комірок, з номерами $1,2,...,q$, а друга − h комірок, з номерами $q+1,q+2,...,H$.

Це означає, що палета ділиться на дві частини (ліву та праву): одна частина палети містить $S \times q$ комірок – місць для завантаження предметів, інша – $S \times h$ комірок. Порядок завантаження палети наведено на рис. 2.

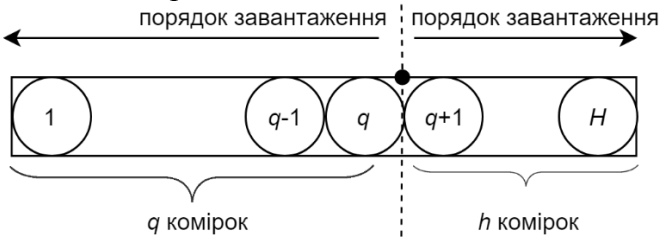


Рис. 2. Порядок завантаження ряду палети

З урахуванням такого порядку завантаження, коміркам можна поставити у відповідність три координати (a,b,c): a – частина ряду (l – ліва, r – права); b – номер ряду ($b=\overline{1,S}$); c – позиція в ряду ($c=\overline{1,q}$ – лівий ряд, $c=\overline{1,h}$ – правий ряд, нумерація позиції починається з точки поділу ряду).

На палету повинно бути завантажено SH предметів, які зберігаються на складі. Комірки в кожному ряді можуть заповнюватися тільки послідовно з фронтальних сторін, починаючи від точки поділу палети. При цьому, c-та комірка кожного ряду (починаючи з другої) може бути заповнена тільки після того, як в її ряді буде заповнена комірка $c-1$. Палету обслуговують m роботів. Для того, щоб помістити необхідний предмет на позицію abc роботу j, необхідно витратити $p_{(abc)j}$ одиниць часу ($a=l,r$; $b=\overline{1,n}$; $c=\overline{1,q}$; $j=\overline{1,m}$). У частковому випадку, коли роботи мають однакову продуктивність, виконується:

$$p_{(abc)j}=p_{(abc)},\ j=\overline{1,m}.$$

Необхідно визначити порядок завантаження палети, при якому ро́боти завантажили б усі комірки за мінімальний час.

Задача складання плану завантаження палети зводиться до наступної задачі теорії розкладу: задано множину робіт $J\left(\left|J\right|=S\cdot H\right)$, що розбиті на n ланцюгів, де $n=2S$. Ланцюг i складається з n_i робіт, $i=\overline{1,n}$. Кожна робота з J описується парою ij, де i – номер ланцюга, j – порядковий номер у ланцюзі, $j=\overline{1,n_i}$, і тривалістю виконання p_{ij}.

Кількість пристроїв, що виконують цю множину робіт – m. Пристрої працюють паралельно і кожний з них може виконувати будь-яку роботу з множини J. Пристрої відрізняються один від одного продуктивністю виконання робіт і є одноманітними (пропорційними). Тобто, пристрої можна впорядкувати за швидкістю виконання робіт і цей порядок однаковий для всіх робіт: для пристрою v заданий коефіцієнт k_v такий, що тривалість виконання роботи ij на пристрої v дорівнює $k_v p_{ij}$, $v=\overline{1,m}$. «Еталонним» будемо називати пристрій з коефіцієнтом продуктивності $k=1$. У цьому сенсі величина p_{ij} є тривалістю виконання роботи ij на еталонному пристрої. Величину k_v будемо називати *коефіцієнтом продуктивності* (якщо $k_v>1$, то пристрій v менш продуктивний, ніж еталонний, якщо $k_v<1$ – більш продуктивний).

Передбачається, що всі роботи множини J надходять одночасно, процес обслуговування кожної роботи протікає без переривань. Усі пристрої працюють без переривань.

Необхідно побудувати такий розклад виконання робіт множини J, при якому досягає мінімуму загальний час виконання робіт: $C_{\max}=\max\limits_{ij}C_{ij}=\max\limits_{i}C_{in_i}\to\min$, де C_{ij} – момент завершення роботи ij, C_{in_i} – момент завершення останньої роботи i-того ланцюга.

Далі, не втрачаючи загальності, будемо вважати, що виконуються такі умови: усі $k_v\geq 1$, $k_1=1$; усі p_{ij} є цілими числами. Задача, що розглядається належить до класу NP.

3. Дослідження властивостей задачі

Нехай $P_i=\sum\limits_{j=1}^{n_i}p_{ij}$, $i=\overline{1,n}$. Упорядкуємо ланцюги по незростанню їх сумарної тривалості робіт $P_1\geq P_2\geq\ldots\geq P_n$, а пристрої за їх продуктивністю так, що $k_1\leq k_2\leq\ldots\leq k_m$, $k=\overline{1,m}$.

Знайдемо нижню оцінку C^* загального часу виконання всіх робіт.

З однієї сторони, мінімально можливе значення C^* відповідає «рівномірному» розкладу, тобто розкладу, в якому кожен з пристроїв закінчує роботу в момент часу

$$\sum_{i=1}^{n}\sum_{j=1}^{n_i}\frac{1}{\sum\limits_{v=1}^{m}\frac{1}{k_v p_{ij}}}.$$

Однак, з урахуванням того, що в даній задачі роботи мають відношення передування у вигляді ланцюгів, то загальний час виконання всіх робіт не може бути меншим ніж час виконання робіт найдовшого з ланцюгів найпродуктивнішим пристроєм; час виконання робіт другого за довжиною з ланцюгів другим за продуктивністю пристрою, тощо. Тобто:

$$C^*=\max\left\{P_1k_1;\ P_2k_2;\ldots;P_mk_m;\sum_{i=1}^{n}\sum_{j=1}^{n_i}\frac{1}{\sum\limits_{v=1}^{m}\frac{1}{k_v p_{ij}}}\right\}. \qquad (1)$$

Величина C^* може бути як цілим так і не цілим числом.

Враховуючи (1), маємо два випадки, що є важливими для аналізу задачі:

1) коли максимум у (1) відповідає останньому елементу;

2) коли максимум у (1) відповідає одному з перших елементів.

Розглянемо деякий розклад σ. Позначимо в цьому розкладі:

$J_v(\sigma)$ – упорядкована множина робіт, що виконується пристроєм v, $v=\overline{1,m}$;

$C_v(\sigma)=\sum\limits_{ij\in J_v(\sigma)} k_v p_{ij}$ – тривалість зайнятості пристрою v, $v=\overline{1,m}$;

$$\Delta_v(\sigma,C^*)=\max\{0;C_v(\sigma,C^*)-C^*\}=$$

$$=\max\left\{0;\sum_{ij\in J_v(\sigma)} k_v p_{ij}-C^*\right\}$$ – *виступ* пристрою v, $v=\overline{1,m}$;

$$R_v(\sigma,C^*)=\max\{0;C^*-C_v(\sigma,C^*)\}=$$

$$=\max\left\{0;C^*-\sum_{ij\in J_v(\sigma)} k_v p_{ij}\right\}$$ – *резерв* пристрою v, $v=\overline{1,m}$.

Визначимо: $I_\Delta(\sigma)$ – множина пристроїв, для яких $\Delta_v(\sigma)>0$; $I_R(\sigma)$ – множина пристроїв, для яких $R_v(\sigma)>0$; $I_0(\sigma)$ – множина пристроїв, для яких $\Delta_v(\sigma)=R_v(\sigma)=0$. Із визначення величин $\Delta_v(\sigma)$ та $R_v(\sigma)$ слідує, що $\Delta_v(\sigma)R_v(\sigma)=0$, $v=\overline{1,m}$.

З урахуванням позначень критерій оптимізації вихідної задачі (мінімізація загального часу виконання робіт) має вигляд:

$$C_{\max}(\sigma)=C^*+\max_{1\le v\le m}\Delta_v(\sigma)\to\min.$$

У свою чергу, з урахуванням того, що $C^*=const$, тобто не залежать від упорядкування, критерій спрощується і зводиться до мінімізації максимального виступу:

$$C_{\max}(\sigma)\to\min \sim \max_{1\le v\le m}\Delta_v(\sigma)\to\min.$$

При побудові розкладів будемо вважати, що момент r запуску пристроїв дорівнює нулю. Рис. 3 ілюструє взаємозв'язок величин, що входять до вказаних критеріїв.

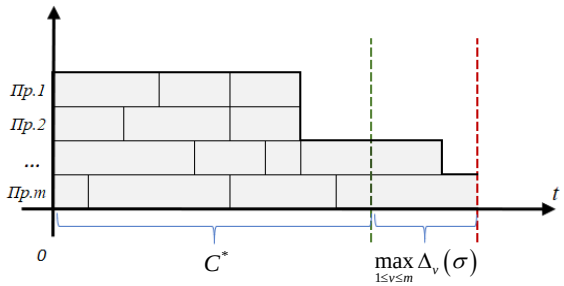


Рис. 3. Взаємозв'язок величин

Застосовуючи методологію побудови ПДС-алгоритмів [6], згідно з якою поліноміальна складова алгоритму породжується логіко-аналітичними умовами, виконання яких гарантує оптимальність отриманого рішення, визначимо достатні умови оптимальності (ДУО) розкладів.

4. Достатні умови оптимальності розкладів

Випадок 1

Якщо $C^*=\sum\limits_{i=1}^{n}\sum\limits_{j=1}^{n_i}\dfrac{1}{\sum\limits_{v=1}^{m}\dfrac{1}{k_v p_{ij}}}$ та $C_{\max}(\sigma)=C^*$, то розклад σ є оптимальним. При цьому C^* – ціле та усі $\dfrac{C^*}{k_v}$ також є цілими числами, для заданої множини робіт отримано рівномірний розклад, у якому:

$$\sum_{v=1}^{m}\Delta_v(\sigma)=\sum_{v=1}^{m}R_v(\sigma)=0.$$

Випадок 2

Якщо $C^*=\max\{P_1k_1;\ P_2k_2;...;P_mk_m\}$ та $C_{\max}(\sigma)=C^*$ то розклад σ є оптимальним. При цьому $\sum\limits_{v=1}^{m}\Delta_v(\sigma)=0$.

5. Алгоритми побудови початкових розкладів

Розроблено чотири евристичних алгоритми побудови початкового розкладу, метою яких є рівномірне навантаження пристроїв. Розглянемо два з них.

Алгоритм *A1* – алгоритм побудови розкладу без простоїв

У даному алгоритмі поточна робота призначається на той пристрій, в якого невикористаний зведений фонд робочого часу найбільший і попередня робота завершена до часу завершення поточної останньої роботи цього пристрою.

Позначення: $J^* \in J$ − множина робіт, в яких усі попередники призначені на виконання.

КРОК 1. Розрахувати C^*.

КРОК 2. Сформувати множину $J^* := \{p_{11};\ p_{21};\ ...;\ p_{n1}\}$.

КРОК 3. Встановити значення величин невикористаного зведеного фонду робочого часу пристроїв: $f_v := \dfrac{C^*}{k_v},\ v = \overline{1,m}$.

КРОК 4. Встановити поточну тривалість зайнятості пристроїв: $T_v := 0,\ v = \overline{1,m}$.

КРОК 5. Обрати серед робіт множини J^*, роботу ij, що має найбільшу тривалість.

КРОК 6. Визначити множину пристроїв $M^* \in M$, на які можна призначити роботу ij: $v \in M^*$, якщо $T_v \geq C_{i,j-1}$ (робота, що передує роботі ij виконана до моменту поточної тривалості зайнятості пристрою).

КРОК 7. Обрати з множини M^* пристрій v із максимальним невикористаним зведеним фондом робочого часу f_v (а якщо таких декілька, то обрати пристрій з найменшим k_v).

КРОК 8. Призначити роботу ij на пристрій v: $T_v := T_v + k_v p_{ij};\ f_v := f_v - p_{ij}$.

КРОК 9. Видалити роботу ij з множини J^*: $J^* := J^* \setminus \{(i,j)\}$.

ЯКЩО $j < n_i$, **ТО** $J^* := J^* \cup \{(i,\ j+1)\}$.

КРОК 10. ЯКЩО $J^* = \varnothing$ **ТО** кінець алгоритму,

ІНАКШЕ перейти на **КРОК 5**.

Алгоритм *А2* – алгоритм побудови розкладу з простоями

Нехай B_{ij} − початок виконання роботи ij.

КРОК 1. Розрахувати C^*.

КРОК 2. Сформувати множину робіт $J^* := \{p_{11};\ p_{21};\ ...;\ p_{n1}\}$.

КРОК 3. Встановити значення величин невикористаного зведеного фонду робочого часу пристроїв: $f_v := \dfrac{C^*}{k_v},\ v = \overline{1,m}$.

КРОК 4. Встановити поточну тривалість зайнятості пристроїв: $T_v := 0,\ v = \overline{1,m}$.

КРОК 5. Обрати серед робіт множини J^*, роботу ij, що має найбільшу тривалість.

КРОК 6. Обрати пристрій v з множини M із максимальним невикористаним зведеним фондом робочого часу f_v (а якщо таких декілька, то обрати пристрій з найменшим k_v).

КРОК 7. Призначити роботу ij на пристрій v:

ЯКЩО $T_v \geq C_{i,j-1}$; **ТО** $B_{ij} := T_v$; $T_v := T_v + k_v p_{ij};\ f_v := f_v - p_{ij}$;

ІНАКШЕ $B_{ij} := C_{i,j-1}$; $T_v := C_{i,j-1} + k_v p_{ij};\ f_v := f_v - (C_{i,j-1} + k_v p_{ij})/k_i$.

КРОК 8. Видалити роботу ij з множини J^*: множини $J^* := J^* \setminus \{(i,j)\}$.

ЯКЩО $j < n_i$, **ТО** $J^* := J^* \cup \{(i,\ j+1)\}$,

ЯКЩО $J^* = \varnothing$ **ТО** кінець алгоритму,

ІНАКШЕ перейти на **КРОК 5**.

Розроблені також модифікації цих алгоритмів, в яких кроки де, для поточної роботи обирається пристрій, містять елементи випадковості (КРОК 7 алгоритму А1 та КРОК 6 алгоритму А2).

Згідно методології побудови ПДС-алгоритмів задача складання розкладу виконання робіт з відношенням передування паралельними пристроями за критерієм мінімізації загального часу виконання робіт, вирішується наступним чином: 1) будується початковий розклад; 2) перевіряється виконання достатніх умов оптимальності; 3) якщо ДУО не задовольняються, то ітераційно виконуються перестановки робіт між пристроями, метою яких є максимальне наближення до виконання достатніх умов оптимальності.

6. Висновок

Досліджено властивості задачі календарного планування виконання множини робіт з відношенням передування паралельними пристроями різної продуктивності з метою побудови розкладу із мінімальним загальним часом виконання робіт. Визначені достатні умови оптимальності розкладів. Розроблено алгоритми побудови початкових розкладів.

Список літератури

1. Сергиенко И.В. Задачи дискретной оптимизации. Проблемы, методы решения, исследования / И.В. Сергиенко, В.П. Шило. – К.: Наукова думка, 2003. – 260 с.
2. Зак Ю.А. Прикладные задачи теории расписаний и маршрутизации перевозок / Ю.А. Зак. – М.: Книжный дом «ЛИБРОКОМ», 2012. – 394 с.
3. C.J. Liao and C.H. Lin, “Makespan Minimization for Two Uniform Parallel Machines”, International Journal of Production Economics, vol. 84, No 2, pp. 205–213, 2003.
4. Senthilkumar P. Review of Single Machine Scheduling Problem with Uniform Parallel Machines / P. Senthilkumar, S. Narayanan // Intelligent Information Management. – 2010. – №2. – P. 457–474.
5. Cheng Zhenmin. An approximate algorithm for parallel machine scheduling problem to minimize total completion time [Текст] / Zhenmin Cheng // Computing & Information Systems. – 2010. – P. 187 - 198.
6. Згуровский, М. З. Принятие решений в сетевых системах с ограниченными ресурсами: монография / М. З. Згуровский, А. А. Павлов. – К.: Наукова думка, 2010.– 573 с.

УДК 004.932.2

ДУШУТІН В.В.
ХІМІЧ О.П.

ОГЛЯД ЗАСТОСУВАННЯ НЕЙРОННИХ МЕРЕЖ ДЛЯ ОБРОБКИ ЗОБРАЖЕНЬ

В даній статті розглянуто способи побудови та організації структур нейронних мереж популярних в машинному навчанні. Порівняно переваги і недоліки різних видів нейромереж. Проаналізовано нейронні мережі таких видів: пряма нейромережа – штучний нейрон (Feedforward Neural Network – Artificial Neuron), само організаційна нейромережа Конохена (Kohonen Self Organizing Neural Network), рекурентна нейромережа (Recurrent Neural Network(RNN)-LSTM), згорткова нейромережа (Convolutional Neural Network), модулярна нейромережа (Modular Neural Network).

In this paper, we consider ways of constructing and organizing the structures of neural networks popular in machine learning. Comparative advantages and disadvantages of different types of neural networks. Neural networks of the following types are analyzed: the Feedforward Neural Network (Artificial Neuron), the Kohonen Self Organizing Neural Network, the Recurrent Neural Network (RNN) - LSTM , Convolutional Neural Network, Modular Neural Network.

1. Введення

Штучна нейронна мережа (ШНМ) є парадигмою обробки інформації, яка була зроблена по типу біологічної нервової системи, такою як мозок, та її здатністю обробляти інформацію. Ключовим елементом цієї парадигми є нова структура системи обробки інформації. Вона складається з великої кількості високо взаємопов'язаних процесорних елементів (нейронів), які працюють у унісон для вирішення конкретних проблем. Нейромережі, як і люди, навчаються на прикладі. ШНМ налаштовується для певної програми, наприклад, розпізнавання образів або класифікація даних, через процес навчання. Навчання в біологічних системах передбачає коригування синаптичних зв'язків, які існують між нейронами. Це також стосується і ШНМ.

Нейронні мережі використовують інший підхід до вирішення проблем, ніж звичайні комп'ютери. У звичайних комп'ютерах використовується алгоритмічний підхід, тобто комп'ютер виконує набір інструкцій для вирішення проблеми. Відомо, що комп'ютери не можуть вирішити проблему, якщо не будуть відмічені конкретні кроки, які комп'ютер повинен виконати. Це обмежує проблему вирішення можливостей звичайних комп'ютерів до проблем, які ми вже розуміємо і вміємо вирішити. Але комп'ютери були б набагато корисніше, якщо б вони могли робити те, що ми робити не вміємо (немає точного алгоритму).

Нейронні мережі обробляють інформацію подібним чином до людського мозку. Мережа складається з великої кількості високо взаємопов'язаних елементів обробки (нейронів), що працюють паралельно для вирішення конкретної проблеми. Нейронні мережі навчаються на прикладі. Вони не можуть бути запрограмовані для виконання конкретного завдання. Приклади слід обережно вибирати, інакше корисний час на навчання буде витрачається даремно або навіть гірше - мережа може працювати неправильно. Недолік полягає в тому, що мережа виявляє, як вирішити проблему сама по собі, її операції можуть бути непередбачуваною.

З іншого боку, звичайні комп'ютери використовують когнітивний підхід до вирішення проблем; те, як проблема повинна бути вирішена, повинна бути відома і зазначена в невеликих однозначних інструкціях. Ці інструкції потім перетворюються на програму високого рівня, а потім на машинний код, який комп'ютер може зрозуміти. Ці машини цілком передбачувані; якщо щось трапиться не так, це пов'язано з несправністю програмного чи апаратного забезпечення.

Нейромережі та звичайні алгоритмічні комп'ютери не конкурують, але доповнюють один одного. Такі завдання більш підходять для алгоритмічного підходу, як арифметичні операції та задачі, які більше підходять для нейронних

мереж. Більш того, велика кількість завдань вимагає систем, які використовують комбінацію двох підходів (звичайно звичайний комп'ютер використовується для нагляду за нейронною мережею) для максимально ефективної роботи.

Одною з найбільш поширеною галуззю застосування нейронних мереж у глибокому навчанні є проблема розпізнавання та класифікації зображень. За останні 5 років через поширення та розвиток штучних нейромереж з'явилося багато нових способів вирішення цієї проблеми. Нижче я спробую оглянути найбільш популярні та використовувані типи нейромереж, а також їхню пристосованість до задачі обробки та класифікації зображень.

2. Пряма нейромережа

Ця нейронна мережа є однією з найпростіших форм ШНМ (штучні нейромережі), де дані або вхід рухаються в одному напрямку. Дані проходять через вхідні вузли та виходять на вихідні вузли. Ця нейронна мережа може мати або не мати прихованих шарів. Данні розповсюджуються хвилею у одному напрямку не впливаючі на попередні вузли. Зазвичай використовується класифікаційна функція активації .

Нижче наведено мережу (рис 1) прямого типу з одним шаром. Тут обчислюється сума вхідних елементів і ваг і подаються на вихід. Вихід враховуються, якщо він перевищує певне значення, тобто порогову величину (зазвичай 0), і нейрон випускається з активованим виходом (зазвичай 1), а якщо він не спрацьовує, то вимикається деактивоване значення (як правило, -1).

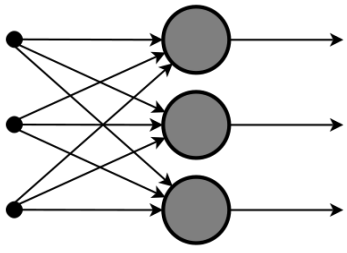


Рис. 1. Структура прямої нейромережі

Прямі нейромережі застосовуються в задачах комп'ютерного зору та розпізнаванні мови, де класифікація цільових класів ускладнюється. Ці нейронні мережі чутливі до шумних даних і легко підтримуються.

3. Самоорганізаційна нейромережа Конохена

Нейромережа Конохена (карта Конохена) – нейронна мережа з навчанням без учителя, яка виконує задачу візуалізації і кластеризації. Метою мережі є введення векторів довільної розмірності до дискретної карти, що складається з нейронів. Карту потрібно навчити, щоб створити власну організацію навчальних даних. Вона може складатися з одного або двох вимірів. Під час тренування карти розташування нейрона залишається постійним, але ваги відрізняються залежно від значення. Цей процес самоорганізації має різні частини, на першій фазі кожне значення нейрона ініціалізується з невеликою вагою та вхідним вектором. На другому етапі нейрон, найближчий до точки, є "виграшним нейроном", а нейрони, пов'язані з виграючим нейроном, також рухатимуться до точки. Відстань між точкою і нейронами розраховується за евклідової відстані, нейрон з найменшою відстані перемагає. Після проведення ітерації всі точки кластеризовані, і кожен нейрон представляє кожен вид кластеру. Це є основою організації роботи нейронної мережі Кохонена.

Нейромережа Конохена використовується для розпізнавання шаблонів даних. Її застосування можна знайти в медичному аналізі для групування даних у різних категоріях. Також карта Конохена може бути використана для розпізнавання шаблонів зображень та їх класифікації.

4. Рекурентна нейромережа

Рекурентна нейронна мережа працює за принципом збереження виходу шару та подачі його назад на вхід, щоб допомогти прогнозувати результат шару.

Рекурентна нейронна мережа (RNN), на відміну від опорної нейронної мережі, є варіантом рекурсивної штучної нейронної мережі, в якій зв'язки нейронів роблять

спрямований цикл. Це означає, що вихід залежить не тільки від поточних входів, але і від стану нейронів попереднього кроку. Ця пам'ять дозволяє користувачам вирішити проблеми з NLP, як розпізнавання рукописного тексту або розпізнавання мовлення.

Довга короткострокова пам'ять (LSTM) - це специфічна рекурентна архітектура нейронних мереж (RNN), яка була розроблена для моделювання тимчасових послідовностей та їх далеких залежностей більш точно, ніж звичайні RNN. LSTM не використовує функцію активації у своїх рекурентних компонентах, збережені значення не змінюються, і градієнт не схильний до зникнення під час тренувань. Зазвичай, блоки LSTM реалізуються в "блоках" з декількома одиницями. Ці блоки мають три або чотири логічні "ворота" (наприклад, вхідні ворота, ворота забувальні, вихідні ворота), які контролюють потоки інформації за допомогою логістичної функції.

В рекурентних мережах перший шар утворюється схожий на пряму нейронну мережу з обчисленням суми ваг і функцій. Процес рекурентної нейронної мережі починається після того, як це обчислюється, це означає, що від одного етапу до наступного кожне нейрон буде пам'ятати деяку інформацію, яку він мав на попередньому кроці. Це робить кожен нейрон схожим на комірки пам'яті у процесі виконанні обчислень. У цьому процесі ми повинні дозволити нейронній мережі працювати “прямим прогоном” і пам'ятати, яка інформація потрібна для подальшого використання. Якщо прогноз нейромережі неправильний, ми використовуємо швидкість навчання або корекцію помилок, щоб внести невеликі зміни, щоб поступово працювати над правильним прогнозуванням під час “зворотнього прогону”.

Застосування повторюваних нейронних мереж можна знайти в моделях перетворення текстового мовлення (TTS). У цій статті описується система "Deep Voice", яка був розроблений в Лабораторії штучного інтелекту Baidu в Каліфорнії. Вона була натхненна традиційною структурою перетворення тексту в мову, замінивши всі компоненти нейронною мережею. По-перше, текст перетворюється на "фонему", а модель синтезу аудіо конвертує її в мову. RNN також реалізується в:.

5. Згорткова нейромережа

Згорткові нейронні мережі (або CNNs) - це особливі види нейронних архітектур, спеціально розроблені для обробки даних зображення. З моменту їх запровадження на початку 1990-х рр. CNNs продемонстрували відмінну продуктивність при таких задачах, як класифікація рукописних цифр та виявлення обличчя. В останні кілька років кілька документів показали, що вони також можуть забезпечити видатні результати для більш складних завдань з візуальної класифікації.

Згорткові нейронні мережі схожі на прямі нейромережі, де нейрони мають здатні до навчання ваги та ухил. ЇЇ застосують в процесі обробки сигналів та зображень, що випереджує OpenCV в області комп'ютерного бачення.

Згорткова нейронна мережа містить один або більше згортальних шарів, шарів ”пулінгу” та повнозвязних шарів (fully connected), і використовує декілька багатошарових персептронов (шарів). Згорткові шари використовують операцію згортки до вхідних даних та переводить результат до наступного шару. Після шару згортки зазвичай розташований шар “пулінгу” . Ця операція дозволяє зменшити данні та поглиблюватися в мережу з набагато меншими параметрами.

Згорткові нейронні мережі демонструють видатні результати у задачах класифікації зображень та мовлення.

CNN застосовуються в таких методах, як обробка сигналів та методи класифікації зображень. Наразі згорткові нейромережі переважають у методах комп'ютерного зору через їх точність у класифікації зображень.

6. Модулярна нейромережа

Модулярні нейронні мережі складаються з сукупність різних мереж, що працюють незалежно та вносять свій вклад у вихідні данні. Кожна нейронна

мережа має декілька входів, які є унікальними в порівнянні з іншими мережами, які будують та виконують підзавдання. Ці мережі не взаємодіють та не сигналізують одна одну про виконання завдань. Перевага модульної нейронної мережі полягає в тому, що вона розбиває великий обчислювальний процес на менші компоненти, що зменшує складність. Це розбиття допоможе зменшити кількість з'єднань і з взаємодії цих мереж один з одною, що, в свою чергу, збільшить швидкість обчислень. Проте час обробки буде залежати від кількості нейронів та їх участі в обчисленні результатів.

Кожна незалежна нейронна мережа служить модулем і працює на окремих входах, щоб виконати певну підзадачу загального завдання, яке виконує мережа. Окрім незалежних нейромереж модуляна нейронна мережа містить у собі модулі посередники. Посередник приймає виходи кожного модуля та обробляє їх, щоб виробляти висновок мережі в цілому. Посередник приймає тільки виходи модулів - він не відповідає, ані іншим сигналом, модулів. Крім того, модулі взаємодіють один з одним.

На відміну від однієї великої мережі, яка може бути призначена для довільних завдань, кожен модуль в модульній мережі повинен бути призначений конкретним завданням і підключений до інших модулів конкретними способами розробником. У деяких випадках дизайнер може вибирати біологічні моделі. В інших випадках інші моделі можуть бути вищими. Якість результату такої нейромережі залежить від якості попереднього її планування та дизайну.

Модулярнні нейронні мережі (MNNs) - це стрімко зростаюче поле в дослідженні штучних нейронних мереж.

7. Висновки

Таким чином, у статті було розглянуто використання різних типів нейронних мереж, їх функції та застосування, а також їх пристосованість для аналізу та класифікації зображень. Для цього було проведено відбір і аналіз найбільш популярних та використовуваних у машинному навчанні типів неромереж , розглянуто їхні властивості та особливості. Було зроблено висновок що, хоча нейромережі являють собою досить універсальні системи, найбільш пристосованими до роботи із зображеннями (computer vision) на данний момент можна вважати згорткові нейромережі (Convolutional neural network). Розробки у галузі модулярних нейромереж (Modular neural network) теж вважаються досить прогресивними та багатообіцяючими.

Список літератури

1. [Електронний ресурс] // Режим доступу: https://arxiv.org/pdf/1702.07825.pdf
2. Sebastian Raschka. Python Machine Learning, 1st Edition–2015.
3. Aurélien Géron. Hands-On Machine Learning with Scikit-Learn and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems 1st Edition–2016.
4. Jeff Heaton.Artificial Intelligence for Humans, Volume 3: Deep Learning and Neural Networks-2015.
5. Tariq Rashid.Make Your Own Neural Network-2016.
6. [Електронний ресурс] // Режим доступу: http://neuralnetworksanddeeplearning.com/

УДК 378.4+004.738.5

КОБЕЦЬ Н.М.,
КОВАЛЮК Т.В.

ЗАДАЧІ СЕНТИМЕНТ-АНАЛІЗУ ЯК ІНСТРУМЕНТ МОНІТОРИНГУ МІРКУВАНЬ КОРИСТУВАЧІВ СОЦІАЛЬНИХ МЕРЕЖ

У даній статті розглянуто методи, задачі та математична модель контент-аналізу. Описаний метод кросмовного аналізу висловлювань та алгоритм категоризації аспектів та визначення рейтингу вислювлювань.

Ключові слова: контент-аналіз, кросмовний аналіз висловлювань,категоризації аспектів

The methods, tasks and a mathematical model of content analysis are considered in this article. The method of cross-language analysis of statements and algorithm of categorization of aspects and determination of rating of impression are described

Key words: content analysis, cross-language analysis of statements, categorization of aspects

1 Вступ

Розвиток глобальної мережі Інтернет і сервісів, які в ній надаються, дає необмежені можливості для збору даних про вибрану предметну область різного характеру, актуальну для соціологів, маркетологів, психологів, лінгвістів тощо. Зокрема, мікроблоги, що дозволяють публікувати в мережі короткі повідомлення за допомогою як стаціонарних, так і мобільних пристроїв, є популярним способом агрегування текстової інформації. Великі компанії використовують соціальні мережі для дослідження думок про свої продукти. Такий підхід, на відміну від використання опитувань на сайті виробника і роботи фокус-груп, забезпечує велику широту дослідження думок. Для вирішення завдань аналізу емоційного забарвлення тексту в комп'ютерній лінгвістиці використовуються методи контент-аналізу, загальна назва для яких – Sentiment Analysis (аналіз тональності тексту).

Отже, задача визначення змісту та емоційного окрасу висловлювань користувачів соціальних мереж є актуальною та важливою.

2 Мета статті

Метою статті є дослідження властивостей задачі контент-аналізу в рамках аспектно-орієнтованого аналізу висловлювань користувачів соціальних мереж.

3. Методи та задачі контент-аналізу тексту

Методи, засновані на аналізі частотних характеристик тексту (frequency-based), для вилучення аспектів застосовують прості фільтри на іменникових конструкції та методи, засновані на відношеннях, які використовують обробку природної мови для пошуку відносин між аспектами та пов'язані з ними почуття. Існують гібридні методи, які використовують відносини для фільтрації аспектів, що часто зустрічаються. Точність гібридних методів значно вище, ніж двох попередніх. Однак, як в двох попередніх випадках, гібридні методи потребують ручного налаштування різних параметрів. Щоб уникнути необхідності ручного налаштування параметрів, використовують методи навчання з учителем, який автоматично вивчає параметри моделі даних. Методи навчання без учителя, а також ймовірнісні моделі дозволяють визначити, про що говориться в тексті, тобто семантику тексту.

4. Математична модель задачі контент-аналізу

Кожне висловлювання можна представити у вигляді п’ятивимірного вектора:

$$\left(e_j, a_{jk}, so_{ijkl}, h_i, t_l\right)$$

де e_j – сутність, для якої виконується аналіз висловлювань;

a_{jk} – один з аспектів сутності e_j;

h_i – автор висловлювання;

t_l – час, коли автор h_i залишив своє висловлювання;

so_{ijkl} – емоційна направленість висловлювання, залишеного автором h_i по відношенню до аспекту a_{jk} сутності e_j в час t_l. Емоція можуть бути позитивною, негативною або нейтральною, або виражати різні рівні інтенсивності від 1 до 5. Пара e_j та a_{jk} завжди виражає ціль висловлювання.

5 Кросмовний аналіз висловлювань

Метод кросмовного аспектно-орієнтованого аналізу висловлювань складається з наступних етапів:

1. Категоризація аспектів сутності, які зустрічаються у висловлюваннях природною мовою в семантичні аспекти.
2. Вилучення висловлювання для різних аспектів
3. Узагальнення кросмовної розбіжності в висловлюваннях для різних аспектів

Розглянемо алгоритм генерування семантичної категорії для *i*-го слова (терміни) pf_i аспекту об'єкту, використовуючи k-тематичну модель.

Вхідні дані: F – кросмовна категоризаційна модель, з поліноміальним розподілом;

$Dirichlet(\alpha)$ – розподіл Діріхлє з параметром α;

pf_i – термін для аспекту об'єкту, використаний у мультимовних рецензіях.

Вихідні дані: $Category(pf_i)$ – семантична категорія pf_i.

Схема алгоритму:

Крок 1. Вилучити кросмовні віртуальні контекстні документи cvd_i для pf_i з багатомовних рецензій.

Крок 2. Підібрати розподіл тем $F(cvd_i)$ з розподілом Діріхлє з параметром α $Dirichlet(\alpha)$.

Крок 3. Для кожного компоненту x_j, який належить вектору cvd_i ($x_j \in cvd_i$), виконати наступне:

Крок 3.1. Підібрати тему $z_j \in \{1, \ldots, K\}$ з $Mult(\Phi(cvd_i))$.

Крок 3.2. Додати тему z_j до множини тем $Topics(cvd_i)$ для поточного документу cvd_i:

$$AddTo(z_j, Topics(cvd_i)).$$

Крок 4. Пов'язати семантичну категорію $Category(pf_i)$ терму pf_i з темою z_k, яка має найвищий ймовірнісним розподіл в $Topics(cvd_i)$:

$$Category(pf_i) \leftarrow z_k.$$

Семантичні категорії pf_i (позначаються як $Category(pf_i)$) генеруються в процесі роботи алгоритму. Тут $Mult(\Phi(cvd_i))$ відноситься до підбору з постеріорного виводу крім тем, наданих в кросмовному віртуальному контекстному документі.

Зі всіма багатовимірними латентними семантичними ключами, вилученими з рецензій, досліджувана категоризаційна модель, може краще визначати розподіл кросмовних семантичних категорій без позначених даних. Навіть, якщо терміни аспектів товару не мають лексичної подібності, але вони знаходяться в одному латентному семантичному контексті, вони можуть мати відносно високу ймовірність опинитися в одній семантичної категорії.

6 Вилучення висловлювань та визначення рівня їх емоційного окрасу

Після проведення кросмовної категоризації аспектів товару, необхідно провести вилучення висловлювань щодо кожної з отриманих категорій та провести аналіз того, наскільки авторам рецензій подобається той, чи інший аспект.

Для підвищення точності результатів аналізу використовуватимемо п'ятирівневу шкалу (від 1 до 5).

Для кожного цільового аспекту запропонований метод спочатку вилучає найближчі синтаксичні одиниці, які служать для вираження почуттів для кожного входження аспекту, що належить до поточної категорії. Для вираження почуттів зазвичай використовується прикметник, який розташований найближче до терміну аспекту в тому самому сегменті речення, який описує якість цього аспекту. Наприклад, для сайту Epinions.com для оцінки рейтингу кожної вилученої одиниці

настрою використовується метод k-найближчих сусідів. За п'ятирівневою шкалою оцінок більшість прикметників має два найближчих сусіда, наприклад, прикметник «дефектний» розміщується між прикметниками «поганий» та «жахливий». Отже далі будемо використовувати коефіцієнт k рівним 2 для методу двох найближчих сусідів в процесі визначення рейтингу аспектів.

Для кожної синтаксичної одиниці настрою будемо використовувати пошук ушир в синонімічному графі словника WordNet з максимальною глибиною 5, щоб знайти два проранжовані синоніми з директиви рейтингу. Потім за допомогою дистанційно-зваженого алгоритму *k*-найближчих сусідів з неперервною цільовою функцією визначимо середньозважені рейтинги двох найближчих сусідів, як рейтинги для синтаксичної одиниці настрою. Так, рейтинг для синтаксичної одиниці настрою *snt* дорівнює:

$$r_{snt} = \frac{\sum_i w_i \times r_i}{\sum_i w_i}$$

де w_i – вага сусіда n_i; r_i – рейтинг сусіда n_i .Розрахунок значення ваги базується на мінімальній відстані між сусідом n_i та синтаксичною одиницею настрою *snt* в ієрархії словника Wordnet :

$$w_i = \frac{1}{dist(snt, n_i)}.$$

Методи, які використовують зважені відстані, дають більшу вагу для ближчих сусідів і є більш стійкими до зашумлених даних.

На рис. 1. показано як проводиться визначення рейтингу «дефектних» аспектів. Нижче подані розрахунки визначають числове значення рейтингу «дефектних» аспектів:

$$w(terrible) = \frac{1}{dist(defective, terrible)} = \frac{1}{4},$$

$$w(poor) = \frac{1}{dist(defective, poor)} = 1,$$

$$r(poor) = 2,\ r(terrible) = 1,$$

$$r(defective) = \frac{\sum_i w_i \times r_i}{\sum_i w_i} = \frac{1\times 2 + \frac{1}{4}\times 1}{\frac{1}{4}+1} = 1.8$$

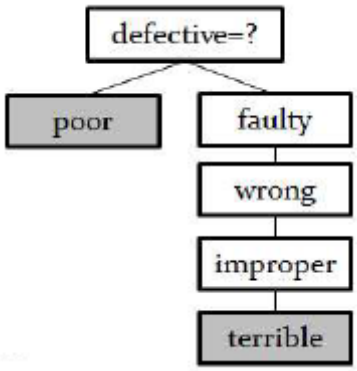


Рис.1 – Визначення рейтингу «дефектних» аспектів.

Нарешті, система агрегує рейтинги усіх синтаксичних одиниць настрою, які виражають ступень задоволення аспектом. Для кожного елемента, виводиться набір типових аспектів і передбачувані рейтинги.

Висновки

Експериментальні результати показують, що досліджувана модель ефективно групує мультимовні назви аспектів в семантичні категорії

Список літератури

1. Andrew Lipsman. Online consumer-generated reviews have significant impact on offline purchase behavior. Technical report, Comscore Inc., 2007. [Електронний ресурс] / Г. Andrew Lipsman. – Режим доступу:

http://www.comscore.com/Insights/Press_Releases/2007/11/Online_Consumer_Reviews_Impact_Offline_Purchasing_Behavior

2. Pang B. Opinion mining and sentiment analysis [Текст]/ Pang B., Lee L // N.Y.:Now Publishers Inc., 2008. – 155 с.

3. Wiebe J. Development and use of a goldstandard data set for subjectivity classifications [Текст] : In Proceedings of the 37th annual meeting of the Association for Computational Linguistics on Computational Linguistics, ACL '99 / Wiebe J., Bruce R., O'Hara T. – Stroudsburg, PA, USA: Association for Computational Linguistics, 1999. – с. 246–253.

УДК 004.852

КРАВЧЕНКО Є. І

РІШЕННЯ З ПРОГРАМНОГО ТА ТЕХНІЧНОГО ЗАБЕЗПЕЧЕННЯ ДЛЯ ФОРМУВАННЯ ТА ПІДТРИМКИ СТАРАП-КОМАНД

У даній статті розглянуто концепцію програмного та технічного забезпечення для формування та підтримки стартап-команд. Визначена бізнес-логіка веб-застосування, засоби розробки та архітектура програмного забезпечення.

Ключові слова: стартап-команда, веб-застосування, бізнес-логіка

This article discusses the concept of software and hardware for creating and maintaining startup commands. The business logic of web application, development tools and software architecture is determined.

Key words: startup team, web application, business logic.

1 Вступ

Починати новий бізнес простіше з підтримкою досвідчених менторів та інвестиціями відомих фондів. У світі працює безліч акселераторів – обмежених за часом програм підтримки стартапів. Команда поміщається в умови, що дозволяють створити проект, здатний вийти на ринок і отримати інвестиції. Отже, створення систем та інструментів підтримки стартап-команд є задачею актуальною.

2 Постановка проблеми

Основа старапа - сама ідея, просування і реалізація якої, приносять прибуток стартаперам. Акцент в даному випадку робиться на незвичайності та ексклюзивності виду бізнесу або його організації. Розробка програмного забезпечення для підтримки діяльності стартап-команд є задачею нетривіальною через потребу врахування в одному інструментарії багатопрофільності задач, що їх потрібно розв’язувати. Серед задач, що вимагають уваги, можуть бути задачі інтелектуального пошуку та аналізу конкурентів стартапу, задачі юридичного профілю для вирішення проблем інтелектуальної власності, задачі фінансового характеру для розрахунку собівартості (витрат) на розробку стартап-пропозиції та визначення прибутку тощо.

3 Мета статті

Мета статті полягає у визначенні архітектури та засобів реалізації програмного забезпечення, а також вимог до технічного обладнання для розгортання веб-застосування підтримки стартап-команд

4 Архітектура програмного забезпечення

Розвиток архітектури клієнт-сервер призвів до створення трирівневої архітектури (three-tiered architecture). Дана архітектура передбачає наявність наступних компонент програми: сервера бази даних, що реалізовує зберігання даних і реалізацію транзакцій, сервера застосування для підтримки бізнес-логіки і клієнтське застосування, що відповідає за представлення даних. (рис.1).

Виділяють такі переваги трирівневої архітектури: високий рівень безпеки, висока надійність, низькі вимоги до швидкості каналу між терміналами і сервером застосувань, низькі вимоги до продуктивності і технічних характеристик, масштабованість і конфігурованість.

Проаналізувавши усі архітектури та їх переваги, проект, що розробляється, будуватиметься за трирівневою клієнт-серверною архітектурою.

5 Бізнес-логіка веб-застосування

Задачі, що мають бути реалізовані в системі підтримки стартап-команд, такі:

- публікація теми проекту та визначення бізнес-ідеї;
- опис вакансій і вимог до кандидатів в команду;
- реєстрація резюме, рекомендацій і мотиваційних листів кандидатів для участі в проекті;
- рейтингування кандидатів та формування команди;

- плануванні стартап-проекту;
- контроль за виконанням завдань проекту;
- формування пропозицій інвестору через бізнес-модель стартап-проекту Canvas;
- пошук інвесторів;
- підтримка зв'язку з потенційними інвесторами

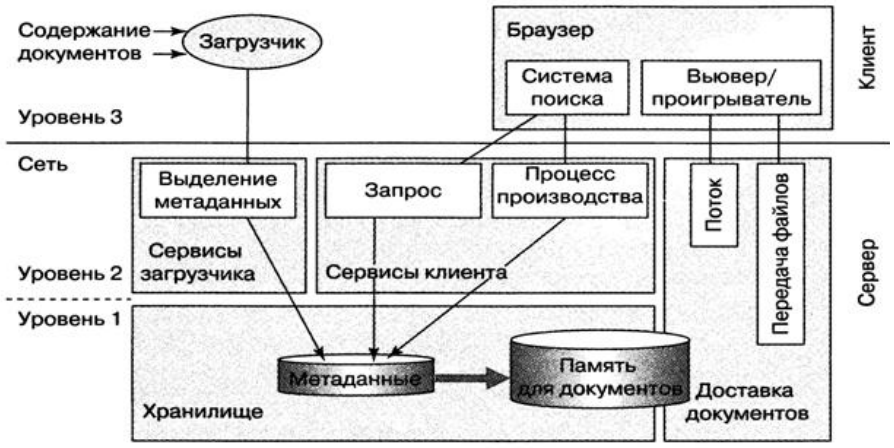


Рис.1 Архітектура трирівневого веб-застосування

6 Засоби розробки

Розробка програмного забезпечення відповідно до його архітектури передбачає реалізацію серверної частини, розробку інтерфейсу користувача та бази даних.

Серверна частина виконана за допомогою об'єктно-орієнтованої мови програмування Java і фреймворку Spring MVC (програмний каркас на основі HTTP сервлета для створення веб-застосувань).

Графічний інтерфейс реалізується мовою розмітки гіпертекстових документів HTML, каскадною таблицею стилів CSS та JavaScript.

У якості бази даних використана MsSQL.

7 Вимоги до технічного забезпечення

Структура технічних засобів визначена, виходячи із можливості забезпечити виконання встановлених операцій. Для правильної роботи серверної частини системи необхідний сервер з характеристиками: процесор з тактовою частотою не нижче 1.6 ГГц; об'єм оперативної пам'яті не менше 2ГБ; об'єм HDD або SDD не нижче 20Гб; доступ до мережі Інтернет.

Для коректної роботи клієнтської частини сервісу необхідні робочі станції з характеристиками: процесор з тактовою частотою не нижче 1,4 ГГц; об'єм оперативної пам'яті не менше 2 ГБ; об'єм HDD або SSD не нижче 40Гб; доступ до мережі Інтернет.

В якості веб-браузера можна використовувати Internet Explorer 10 або вище; браузери, що використовують движок Gecko 27.0 або вище; браузери, що використовують движок Trident 6.0 або вище; браузери, що використовують движок Webkit 537.31 або вище; браузери, що використовують движок Blink 537.11 або вище.

Висновок

Для розробки веб-застосування з підтримкам стартап-команд сформульовані функціональні та нефункціональні вимоги. Визначені засоби розробки та технічне забезпечення для розгортання програмного забезпечення.

Список літератури

1. Tsung-Yi Lin. Focal Loss for Dense Object Detection, 2018. [Електронний ресурс]. Режим доступу: https://arxiv.org/pdf/1708.02002.pdf

2. Інформаційні системи я реалізація інформаційних технологій. [Електронний ресурс]. Режим доступу:http://dn.khnu.km.ua/dn/k_default.aspx?M=k1007&T=04&lng=1&st=0.

УДК 629.7.021:528.831.1:519.652

ЛИСЕНКО А. Р.,
СТАНКЕВИЧ С. А.,
ЛУБСЬКИЙ М.С.,
ЖДАНОВА О. Г.

МОДЕЛЬ АЕРОЗНІМАННЯ З НАДРОЗРІЗНЕННІСТЮ З ЛЕГКОГО БЕЗПІЛОТНОГО ЛІТАЛЬНОГО АПАРАТА НА ОСНОВІ СПІЛЬНОЇ ОБРОБКИ КІЛЬКОХ ЗОБРАЖЕНЬ

Недостатність просторової розрізненності одержуваних аерознімків – поширена проблема. Запропоновано підхід до підвищення просторової розрізненності шляхом поєднання декількох зображень зміщених одне відносно одного, та відновлення нового зображення підвищеної розрізненності. Подано демонстраційний приклад підвищення просторової розрізненності тестових аерозображень цифрової 4К камери зі спектрометром STS Developers Kit, встановленої на борту квадрокоптера DJI Phantom 3

Aerial photograph spatial resolution insufficiency is a common problem. Proposed approach for spatial resolution augmentation by several images, aligned with respect to another, unification and new augmented spatial resolution image recovery. Given demonstrative example of test aerial photo spatial resolution augmentation, taken with 4K camera with STS Developers Kit sprectrometer wich is established aboard DJI Phantom 3 quadrapter

Ключові слова: аерознімання, субпіксельна обробка, підвищення просторової розрізненності

1. Вступ

На сьогодні аерозйомка є надзвичайно поширеною: вона застосовується в будівництві, військовій промисловості, картографії, рекламних цілях, екології, науці та просто як хоббі. Для задоволення потреб будь-якої з перелічених вище категорій використовуються різні технічні засоби, в тому числі і квадрокоптери, які набули великої популярності за їх можливості та відносну дешевизну. Однією з головних характеристик будь-якого зображення є його просторова розрізненність, підвищення якої вимагає потужної апаратури. У зв'язку з розмірами квадрокоптера постає проблема підвищення просторової розрізненності зображення, яка полягає, крім підвищення вартості квадрокоптеру при встановленні додаткових фотоелементів, так і в фізичних обмеженнях носія оптичної системи [1].

Вирішенням даної проблеми може бути субпіксельна обробка послідовності зображень, одержуваних з квадрокоптера [2].

2. Попередні дослідження

Головна ідея полягає в отриманні кількох послідовних зображень геометрично зміщених одне відносно одного на певну частку піксела по рядках та стовпцях, після чого за допомогою їх спільної обробки відновлюється єдине результуюче зображення підвищеної розрізненності. Це досягається завдяки переходу від індивідуальної піксельної сітки кожного окремого зображення, до спільної субпіксельної сітки, яка їх поєднує [3].

3. Відновлення зображень

Нехай модель зображення має наступний вигляд:

$$g(x,y) = h(x,y) * f(x,y) + n(x,y),$$

де $h(x,y)$ – це функція розподілу точок, $f(x,y)$ – вхідне зображення, $g(x,y)$ – вихідне зображення та $n(x,y)$ – шум.

Перетворенням Фур'є даної моделі є:

$$G(u,v) = H(u,v) * F(u,v) + N(u,v)$$

Задача відновлення розрізнювальної здатності полягає в аналітичному подовженні сигналу до $F(u,v)$ за допомогою попередньої інформації про об'єкт і певної технології пост-обробки та отриманні нової функції розподілу точок $H^{'}(u,v)$.

Отже, треба здійснити розрахунок субпіксельних значень x_{ij}, $i=\overline{1,n_x+p-1}$, $j=\overline{1,n_y+q-1}$, де n_x – кількість пікселів осі x, n_y – кількість пікселів по осі y, p – субпіксельне зміщення по осі x, а q – субпіксельне зміщення по осі y відповідно.

Значення пікселів виражаються через таку систему рівнянь:

$$y_{ij}=\sum_{k=1}^{i+p-1}\sum_{l=1}^{j+q-1}x_{kl},\ i=\overline{1,n_x},\ j=\overline{1,n_y}. \quad (1)$$

Маємо систему з $n_x * n_y$ рівнянь та $(n_x+p-1)*(n_y+q-1)$ невідомими.

Розв'язком системи (1) являється рекурентна формула:

$$x_{ij}=y_{(i-p+1)(j-q+1)}-\sum_{k=i-p+1}^{i-1}\sum_{l=j-q+1}^{j-1}x_{kl}-\sum_{k=i-p+1}^{i-1}x_{kj}-\sum_{l=j-q+1}^{j-1}x_{il},$$

$$i=\overline{p,n_x+p-1},\ j=\overline{q,n_y+q-1}.$$

Систему (1) можна переписати у матричну форму:

$$T_1\times X\times T_2=Y, \quad (2)$$

де $X=\{x_{ij}\}$ – матриця невідомих розмірності $(n_x+p-1)\times(n_y+q-1)$, $Y=\{y_{ij}\}$ – матриця вхідних зображень розмірності $n_x\times n_y$, T_1 – p-діагональна матриця розмірності $n_x\times(n_x+p-1)$ з одиницями на діагоналях, T_2 – q-діагональна матриця розмірності $n_y\times(n_y+q-1)$ з одиницями на діагоналях.

Якщо n_x+p-1 є кратним p, а n_y+q-1 є кратним q, тоді розв'язком системи (2) з найменшою вибірковою дисперсією матриці X є:

$$X=pinvT_1\times Y\times pinvT_2,$$

де $pinv$ – псевдообернення матриці T [4].

Модель з шумом, в даному випадку, має наступний вигляд:

$$X=R\times Y+\varepsilon,$$

де X – зображення підвищеної розрізненності у векторній формі, Y – кортеж зображень низької розрізненності у векторних формах, R – матриця пересемплювання, ε – вектор похибок.

Найкраща лінійна оцінка зображення підвищеної розрізненності X за Y має вигляд:

$$X=\operatorname{cov}X\times R^T\times(R\times\operatorname{cov}X\times R^T+\operatorname{cov}\varepsilon)^{-1}\times(Y-meanY)+meanX, \quad (3)$$

де $\operatorname{cov}X$ та $\operatorname{cov}\varepsilon$ – апріорні коваріаційні матриці зображення підвищеної розрізненності та похибок, $meanX$ та $meanY=RmeanX$ – математичні очікування зображення підвищеної розрізненності та вхідних зображень [5]. Для того, щоб розв'язок (3) відповідав максимуму апостеріорної імовірності необхідно, щоб розподіли X та ε були гаусівськими.

Якщо зображення підвищеної розрізненності та похибки є стаціонарними у широкому розумінні, то матриці $\operatorname{cov}X$ та $\operatorname{cov}\varepsilon$ визначаються належними автоковаріаційними функціями, до того ж матриця є блоково-діагональною, так як похибки різних зображень низької розрізненності є некорельованими. Тоді розв'язок може бути отримано за допомогою гаусівської регуляризації [6].

4. Розрахунок стохастичних субпіксельних зсувів зображень

Так як на процес отримання зображень квадрокоптером впливають стохастичні збурення, виникає необхідність оцінювання субпіксельних зсувів отриманих знімків.

Нехай X – матриця першого зображення низької роздільної здатності розмірності $m\times n$, а Y – матриця другого зображення низької роздільної здатності розмірності $m\times n$, де x_{ij} та y_{ij} значення піксела в i-му рядку та j-му стовпцю матриці X та Y відповідно.

Означимо необхідні матриці D, E та F розмірностей $(m-2)\times(n-2)$:

$$d_{ij}=x_{lk}+y_{lk}-x_{(l-2)k}-y_{(l-2)k},$$

$$e_{ij}=x_{(l-1)(k+1)}+y_{(l-1)(k+1)}-x_{(l-1)(k-1)}-y_{(l-1)(k-1)},$$

$$f_{ij} = y_{(l-1)k} - x_{(l-1)k},$$
$$i = \overline{1, m-2},\ j = \overline{1, n-2},\ l = \overline{3, m},$$
$$k = \overline{2, n-1};$$

Тоді субпіксельний зсув можна розрахувати як $C = \frac{A}{B}$, де A розмірності 3×3 має вигляд:

$$A = \begin{bmatrix} (D \circ D)_{середнє} & (D \circ E)_{середнє} & D_{середнє} \\ (D \circ E)_{середнє} & (E \circ E)_{середнє} & E_{середнє} \\ D_{середнє} & E_{середнє} & 1 \end{bmatrix},$$

де $X \circ Y$ – добуток Адамара, а $X_{середнє}$ – середнє арифметичне матриці X.

А матриця B розмірності 3×1 має вигляд:

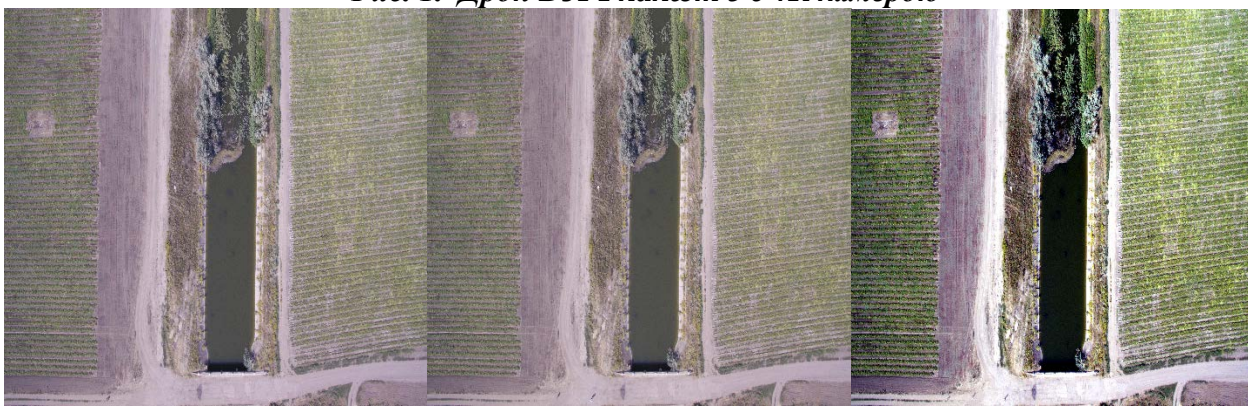
$$B = \begin{bmatrix} (F \circ D)_{середнє} \\ (F \circ E)_{середнє} \\ F_{середнє} \end{bmatrix}.$$

Тоді $u = c_{11}$ – відповідає за зміщення зображень одне відносно одного, а $v = c_{21}$ – відповідає за їх поворот.

5. Результат

Даний метод відновлення субпіксельно реєстрованих зображень досліджувався на реальних аерознімках (рис. 2), одержаних за допомогою 4К камери на основі спектрометру STS Developers Kit встановленої на дроні DJI Phantom 3.

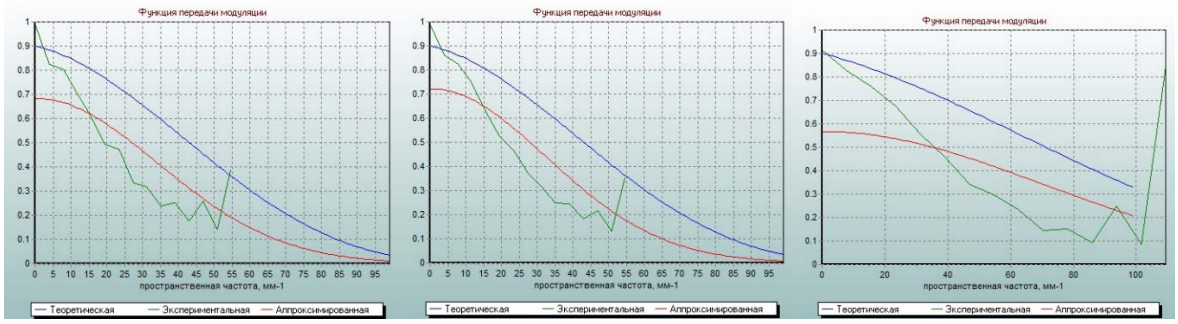
Рис. 1. Дрон DJI Phantom 3 з 4К камерою



а *б* *в*

Рис. 2. Тестові аерозображення:
а, в – вхідні аерозображення низької розрізненності,
в – відновлене аерозображення підвищеної розрізненності

За допомогою функцій передачі модуляції (ФПМ) була отримана оцінка просторової розрізненності вхідних та відновленого зображень. Функції були експериментально визначені за допомогою спеціального програмного забезпечення [7]. Результати оцінок ФПМ наведено на рис. 3.



а *б* *в*

Рис. 3. Функції передачі модуляції тестових аерозображень: а, б – вхідних зональних зображень низької розрізненності, в – відновленого аерозображення підвищеної розрізненності

Як видно з графіків на рис. 2, значення ФПМ на рівні 0,3 відповідають розрізненності 32 мм$^{-1}$ для першого зображення (рис. 3*а*), 31 мм$^{-1}$ для другого зображення (рис. 3*б*) та 54 мм$^{-1}$ для відновленого зображення, що є близьким до теоретичної 60 мм$^{-1}$. Отже, просторова розрізненність відновленого зображення краща приблизно на 70% за вхідні зображення.

6. Висновок

За допомогою даного методу було продемонстровано можливість підвищення просторової розрізненності зображень, але він все-ж має недоліки.

По-перше, великі обчислювальні затрати, які степенево залежать від розмірів вхідних зображень, унеможливлюють роботу даного алгоритму в режимі реального часу. Тому варто розглянути більш обчислювально економічні, і в той же час ефективні, методи відновлення.

По-друге, через стохастичні збурення неможливо досягти строго лінійного субпіксельного зміщення зображень. Тому потрібно вживати певні заходи для просторової і кутової стабілізації зображень, одержуваних на квадрокоптерах.

Разом з тим, даний підхід до підвищення просторової розрізненності є досить перспективним та вартим подальших досліджень та експериментів.

Література

1. Neumann K.J. Trends for digital aerial mapping cameras // The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, 2008.– Vol.XXXVII.– Part B1.– P.551,554.
2. Станкевич С.А., Шкляр С.В., Лубський М.С. Підвищення просторової розрізненності аерознімання на основі субпіксельної реєстрації зображень // Збірник наукових праць Державного науково-дослідного інституту авіації, 2013 – Вип. №16 – УДК 528.831.1, ст. 110-117
3. Liu H.Y., Zhang Y.S., Ji S. Study on the methods of super-resolution image reconstruction // The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, 2008.– Vol.XXXVII.– Part.B2.– P. 461 , 466.
4. Stankevich S.A., Shklyar S.V., Tyagur V.M. Subpixel resolution satellite imaging technique // Proceedings of the 9th International Conference on Digital Technologies (DT'2013).– Žilina: University of Žilina, 2013.– P.81, 84.
5. Ben,Israel A., Greville T.N.E. Generalized Inverses: Theory and Applications. – New York: Springer,Verlag, 2003. – 420 p.
6. Rao C.R. Linear Statistical Inference and Its Application.– N.Y.: John Wiley, 2002. – 656 p.
7. Станкевич С.А., Шолонік О.В. Інструментарій оцінювання еквівалентної просторової розрізненності багато- та гіперспектральних цифрових аерокосмічних знімків // Збірник наукових праць Державного науково-дослідного інституту авіації, 2007.– Вип.3(10).– С.165-171.

УДК 681.007.05

ОСИПЕНКО О.А.

АНАЛІЗ СУЧАСНИХ РІШЕНЬ ТА ЇХНЬОЇ ПРОДУКТИВНОСТІ ДЛЯ ВИРІШЕННЯ ЗАДАЧІ ВИЯВЛЕННЯ ТА РОЗПІЗНАВАННЯ ВІЗУАЛЬНИХ ОБ'ЄКТІВ

В даній статті розглянуті сучасні рішення та проведено аналіз продуктивності систем, які розробляються в розділі комп'ютерного зору для вирішення задачі виявлення та розпізнавання об’єктів. Визначено особливості різних систем, описана необхідність вирішення задач класифікації та застосування даних систем візуального розпізнавання в сучасності.

This article analyzes modern solutions and the performance of systems which are developed in the computer vision section to solve the problem of detecting and recognizing objects. The features of different systems are determined, the necessity of solving the problems of classification and application of these visual recognition systems in the present is described.

1. Вступ

У наш час розпізнавання образів – одна з найскладніших задач, вирішення якої надає людству величезну кількість можливостей. Зараз генерується багато візуальної інформації: фотографії, відео, і у сфері інформаційних технологій важливе місце посідає створення та розвиток систем, які допомагають розпізнавати об'єкти у реальному часі. Цими задачами займається розділ «Комп’ютерний зір» у сфері штучного інтелекту. Крім аналізу зображень, комп'ютерний зір також включає в себе виявлення подій, розпізнавання об'єктів, навчання, відновлення зображень і відстеження візуального потоку даних. За останні роки саме задача виявлення та розпізнавання візуальних об’єктів була причиною створення та швидкого розвитку розділу комп’ютерного зору. При чому на початкових етапах у 2001 році можливо було розпізнати людину лише людину і лише на рівній неперевернутій фотографії. Зараз же сучасні системи працюють з великою кількістю об’єктів і з великим об’ємом вхідної візуальної інформації.

Можна також зазначити, що розпізнавання об'єктів у режимі реального часу має вирішальне значення для багатьох застосувань безпілотних повітряних апаратів, а це дуже актуально через військові дії на сході України.

2. Постановка проблеми

Не дивлячись на розвиток систем «Комп’ютерного зору», в даний час вирішення задачі виявлення та розпізнавання візуальних об’єктів вимагає великих обчислювальних ресурсів. І саме це є однією з найбільших проблем на шляху розвитку цієї галузі. Для аналізу продуктивності таких систем існують показники:

- кадрів в секунду (frames per second) – кількість зображень з вхідного потоку, що може бути оброблена системою за одну секунду;
- об’єм використовуваної оперативної пам’яті;

3. Мета статті

Мета статті - порівняння багатьох моделей для візуального розпізнавання об'єктів та визначення найбільш ефективного, з вказанням переваг , недоліків та додаванням порівняльних таблиць до викладеного матеріалу.

Для цього необхідно вирішити такі задачі:

- провести аналіз сучасних систем, які вирішують задачу виявлення та класифікації з різними математичними моделями у своїх основах;
- провести порівняння цих систем за зазначеними показниками.

4. Система виявлення та розпізнавання візуальних об’єктів

В даний час однією з найсучасніших систем є згорткова нейронна мережа YOLO, для якої розроблена велика кількість моделей. Одні з найефективніших це: SSD, Retinanet, DSSD. Далі розглянемо їх та їхні архітектури задля аналізу їхньої продуктивності.

5. SSD

SSD базується на нейронній мережі прямого поширення яка створює межі, що огортують об’єкти та оцінює ймовірність, що об’єкт в цих межах відноситься до визначеного класу. Початкові шари нейронної мережі базуються на стандартній архітектурі для класифікації високоякісних зображень.

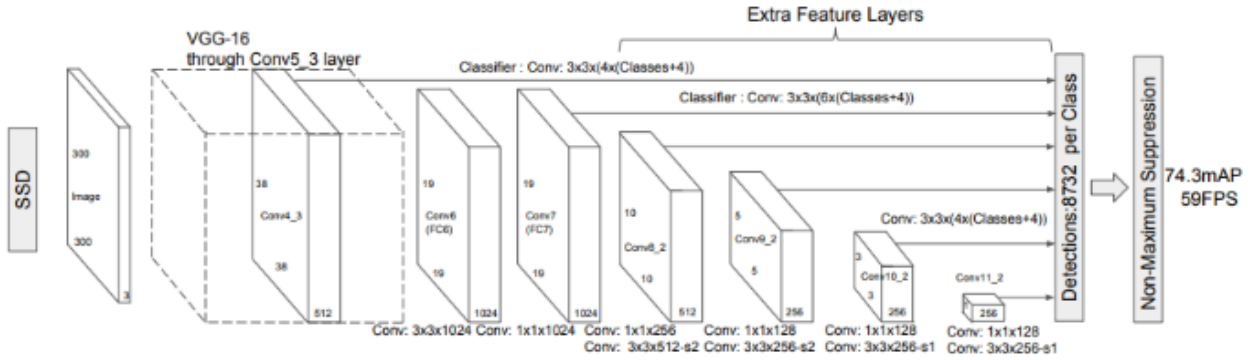


Рис. 1. Архітектура моделі SSD

6. Retinanet

RetinaNet - це єдина уніфікована мережа, що складається з магістральної мережі і двох підмереж, призначених для конкретних завдань. Основа відповідає за обчислення карти згорткового зображення по всьому вхідному зображенню і є згортковою нейронною мережею. Перша підмережа виконує класифікацію згорткових об'єктів на виході базової лінії; друга підмережа виконує згорткову регресію меж, що огортають об'єкти. Найбільш точні детектори об'єктів на сьогодні базуються на двостадійному підході, який популяризується R-CNN, де класифікатор застосовується до малого набору об'єктів, які використовуються для вибору об'єктів. На відміну від цього, одноступеневі сповіщувачі, які застосовуються за допомогою регулярної щільної вибірки місць можливого розміщення об'єктів, мають потенціал бути швидшими і простішими, але досі не досягли достовірності двоетапних детекторів. Результати показують, що RetinaNet здатна відповідати швидкості одноступеневих детекторів, перевершуючи точність всіх існуючих найсучасніших двоступеневих детекторів.

7. DSSD

При пошуку способів подальшого підвищення точності розпізнавання та класифікації очевидними цілями є додавання більшого контексту, особливо для невеликих за розміром об'єктів, на додаток до поліпшення просторового дозволу процесу прогнозування обмежування меж, що обгортують об'єкти . Попередні версії SSD були засновані на мережі VGG , але багато дослідників домоглися більшої точності для задач з використанням Residual-101 . Розглядаючи паралельні дослідження, була проведена робота по інтеграції контексту з використанням так званих мереж «кодувальник-декодер», де звужуючий шар в середині мережі використовується для

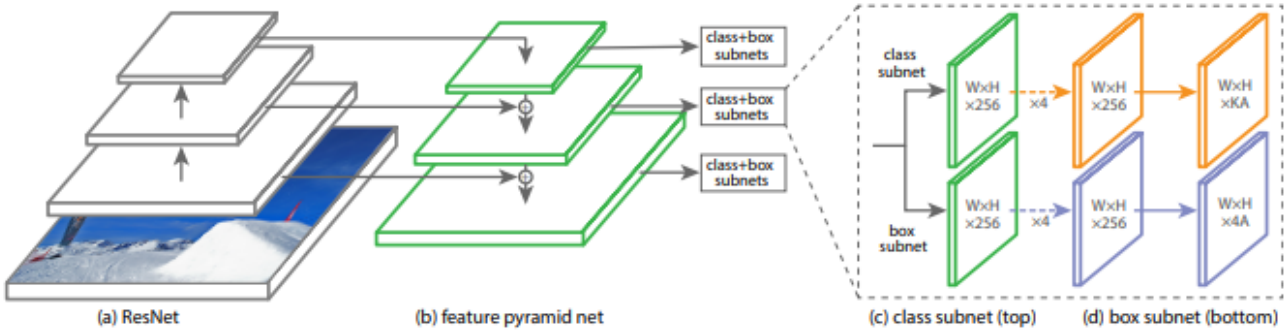


Рис. 2. Архітектура моделі Retinanet

кодування інформації по вхідному зображенню, а потім поступово наступні шари декодують її. Ці підходи також корисні в задачах семантичної сегментації і класифікації пози людини.

$Mavg$ – середнє використання оперативної пам'яті системою протягом тестування, гб.

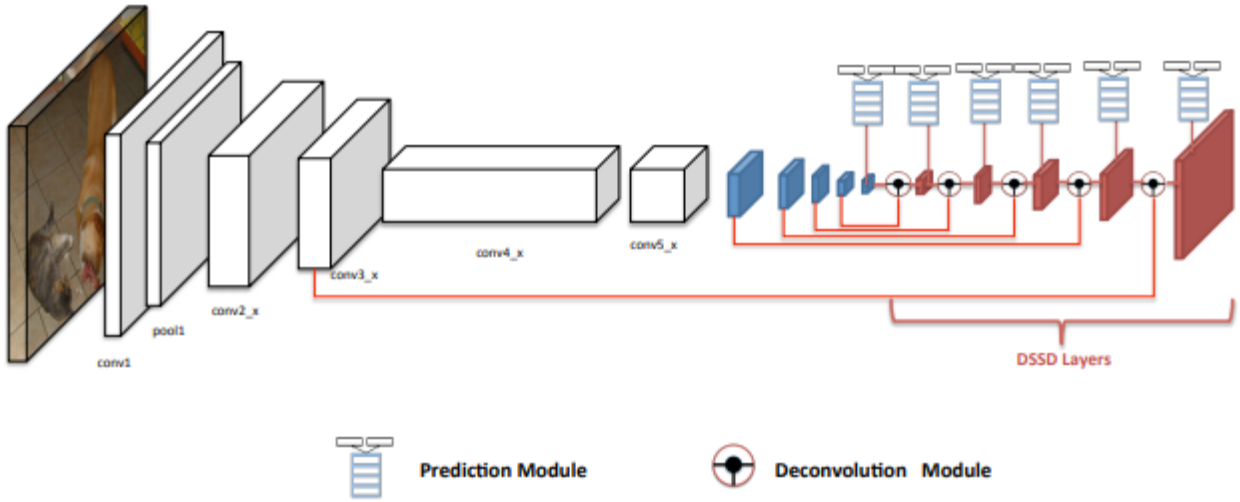


Рис. 3. Архітектура моделі DSSD

8. Математична модель

Маючи три системи із різними моделями нейронних мереж, необхідно сконструювати математичну модель тестування, що надасть можливість нам перевірити їхню продуктивність по замірюваних показниках, а саме кількості кадрів в секунду, що обробляє система, та об'єм використовуваної оперативної пам'яті системою. Представимо параметри кожного тестування у вигляді вхідного вектора T

$$T = (N, M, I, C),$$ де

N – ширина вхідного зображення, пікселів

M – висота вхідного зображення, пікселів

I – кількість вхідних зображень в секунду, що надходять в систему

C – загальна кількість зображень, що беруть участь у тестуванні

та вихідного вектора R

$$R = (Iavg, Mavg),$$ де

$Iavg$ – середня частота кадрів в секунду протягом тестування

9. Тестування

Будемо проводити тестування моделей нейронних мереж у системах, що вже були навчені за допомогою набору даних COCO.

Табл. 1. Результати тестів SSD

Вектор T	Вектор R
(1024,1024,30,30000)	(24.2, 1.2)
(1024,1024,20,30000)	(24.5, 1.1)
(2048,2048,20,30000)	(17.2, 2.2)
(4096, 4096,20,30000)	(17.2, 2.2)

Табл. 2. Результати тестів Retinanet

Вектор T	Вектор R
(1024,1024,30,30000)	(19.3, 0.9)
(1024,1024,20,30000)	(21.3, 0.9)
(2048,2048,20,30000)	(17.6, 2.2)
(4096, 4096,20,30000)	(13.2, 2.8)

Табл. 3. Результати тестів DSSD

Вектор T	Вектор R
(1024,1024,30,30000)	(29.2, 0.6)
(1024,1024,20,30000)	(30.0, 0.6)
(2048,2048,20,30000)	(29.2, 0.9)
(4096, 4096,20,30000)	(27.5, 1.1)

Висновки

За результатами нашого дослідження (табл. 1-3) можна сказати, що моделі, які ми використовували, гарно впорались з аналізом вхідного потоку інформації. Модель Retinanet (табл. 2) при аналізі дала найгірші результати, а DSSD – найвищі (табл.3). Також можна побачити, що вже в даний час нейронні мережі можуть у реальному часі обробляти вхідний потік візуальних даних. Проте, у деяких випадках було б доречно зменшити кількість кадрів в секунду до 20-25, що гарно впливає на продуктивність, і залишає достатньо візуальної інформації для аналізу.

Список літератури

1. Tsung-Yi Lin. Focal Loss for Dense Object Detection, 2018. [Електронний ресурс] https://arxiv.org/pdf/1708.02002.pdf
2. Wei Liu. Single Shot MultiBox Detector. [Електронний ресурс] https://www.cs.unc.edu/~wliu/papers/ssd.pdf
3. Cheng-Yang Fu. Deconvolutional Single Shot Detector, 2017. [Електронний ресурс] https://arxiv.org/pdf/1701.06659.pdf

УДК 57.087.1

МАЛИНОВСЬКИЙ А.Д.,
СПЕРКАЧ М.О.

ЗАДАЧА ПЕРЕВІРКИ ЯКОСТІ СИГНАЛІВ ЕЛЕКТРОКАРДІОГРАМИ ОТРИМАНИХ ЗА ДОПОМОГОЮ ОДНОКАНАЛЬНОГО КАРДІОГРАФА

Розглядається задача перевірки якості сигналів електрокардіограми отриманих за допомогою одноканального кардіографа. Приводиться класифікація існуючих методів перевірки якості сигналу електрокардіограми. Розроблено метод перевірки якості електрокардіограми знятої з одноканального кардіографа на основі алгоритму класифікації за допомогою згорткової нейронної мережі та перевірки наявності контакту. Проведено аналіз отриманих результатів порівняння розробленого методу та методу перевірки якості електрокардіограми на основі вирішальних правил.

The task of checking the quality of electrocardiogram signals obtained with a single-channel cardiograph is considered. The classification of existing methods for checking the quality of the electrocardiogram signal is given. The method of checking the quality of an electrocardiogram record from a single-channel cardiograph on the basis of the classification algorithm with the help of a convolutional neural network and checking the presence of a contact has been developed. The analysis of the obtained results of the comparison of the developed method and the method of checking the quality of the electrocardiogram based on decisive rules is carried out.

Ключові слова: електрокардіограма, кардіограф, згорткові нейроні мережі, задача класифікації

1. Введення

Сучасна медицина рухається в напрямку 3P Medicine (персоналізована, превентивна, предиктивна медицина), в основі якої лежить ідеологія принципу “запобігати, а не лікувати”. Саме тому, набувають популярності носимі девайси, що дозволяють проводити достатньо складні дослідження біометричних показників організму, не лише перебуваючи в медичних закладах, а й у домашніх умовах.

У результаті таких досліджень можна отримати: електрокардіограму, електроенцефалограму, показники кров’яного тиску тощо. Але виникає необхідність у перевірці достовірності та якості отриманих результатів досліджень, що можуть вплинути на виявлення симптомів захворювань людини.

2. Постановка задачі

Одним із досліджень біометричних показників організму людини є зняття електрокардіограми (ЕКГ). Вона являє собою криву з п'ятьма зубцями, що позначаються буквами латинського алфавіту – *P*, *Q*, *R*, *S*, *T* (іноді зустрічається зубець *U*) (рис. 1). Кожен зубець відображає певну стадію збудження міокарда: зубець *P* виникає при порушенні передсердь, комплекс *QRS* – скорочення (систола) шлуночків, зубець *T* з'являється при виході міокарда зі стану збудження. Вид ЕКГ змінюється в залежності від місця її фіксації (відведення). При наявності патологічних змін змінюються висота та ширина зубців, їх співвідношення, періодичність і т.д. Все це враховує кардіолог або лікар функціональної діагностики, коли проводить розшифровку кардіограми і дає свій висновок [1].

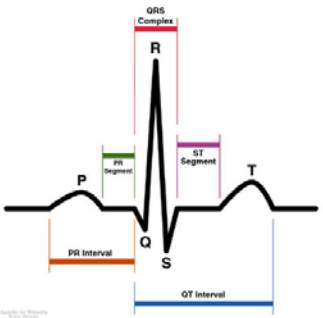


Рис. 1. Основний паттерн кардіограми(один удар серця)

З точки зору аналізу кардіограми за допомогою автоматизованих методів, можливо отримати такі види сигналів:

1) *Сирий сигнал* – вихідний сигнал, знятий безпосередньо з людини, може використовуватись для біометричної ідентифікації, оскільки є найбільш інформативним сигналом [2].

2) *Інтервали між ударами серця* (*RR*-інтервали) – під цим терміном розуміють ряд інтервалів часу між сусідніми *R*-піками кардіограми. Також, *RR*-інтервали можливо вимірювати, за допомогою фотоплетизмографії (просвічування шкіри) [3]. Широко використовується для аналізу модуляцій нервової системи на серці [4].

3) *Анотовані дані* –це координати основних ключових точок удару серця. До таких точок відносяться – точки пов'язані з *P*-хвилею (початок, максимум, кінець), точка *N, Q, R, S,*), точки пов'язані з хвилею *T* [5].

Якість отриманих даних може залежати від впливу багатьох факторів, таких як:

- пози людини – при різних позах, сигнал по різному відображається;
- впливу хімічних речовин [6].

З урахуванням наведених вище теоретичних відомостей про зняття сигналів електрокардіограми сформулюємо **постановку задачі** для перевірки якості таких сигналів.

Отже, людина проводить вимірювання кардіограми за допомогою портативного наручного кардіографа, без участі лікаря. На основі цієї кардіограми будується аналітика стану здоров'я людини.

Завданням на етапі збору сигналу є перевірка його стану та віднесення до одного з наведених класів: «якісний» сигнал і «неякісний» сигнал.

Важливим етапом перевірки якості сигналу та віднесення його до одного з класів є перевірка кількості відрізків на яких відбувається конкатенація сигналу. Під конкатенацією розуміється поєднання сигналів ЕКГ з різних проміжків часу. Така ситуація виникає, коли людина відриває руку від датчика, а потім знову прикладає. Через архітектурні особливості пристрою, за відсутності контакту, припиняється відправка сигналу на смартфон, але одночасно з тим надсилається сигнал, про те, що сигнал відсутній. Ситуація конкатенації призводить до створення ударів серця низької якості, особливо знижується якість *RR*-інтервалу, що виникає на місці склеювання кардіограми. *RR*-інтервали є важливими елементом аналізу ЕКГ.

Після зняття ЕКГ, знаходяться *R*-піки, по яким визначаються ряд *RR*-інтервалів, на основі яких і будується аналіз. Конкатенація сигналу створює похибку в отриманих *RR*-інтервалах. На Рис. 2 наведено приклад *RR*-інтервалу.

Проблема з відривами пальців полягає в тому, що після прикладання знову, сконкатенований удар серця, з точки зору експертної оцінки якості, може бути навіть однозначно класифікований як "якісний".

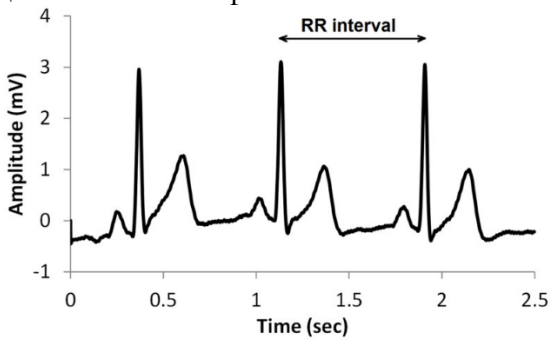


Рис. 2. RR-інтервал

Такий удар серця може вносить неточності в аналіз *RR*-інтервалів. Тому, необхідно контролювати кількість сконкатенованих ударів серця, щоб статистично, такими ударами можна було знехтувати.

3. Класифікація існуючих методів

Підходи до перевірки якості ЕКГ сигналу можна класифікувати на групи, за такими основними критеріями:

1) *Кількість відведень у кардіографа*, для якого будується алгоритм (в залежності від кількості відведень, підходи можуть архітектурно відрізнятись). Наприклад, при перевірки якості сигналу 12-ти канального кардіографа, можливо використовувати підходи, що вирішують задачу класифікації сигналу, за допомогою перевірки сигналів з різних груп відведень, всередині групи;

2) *Тип даних для перевірки.* Підходи перевірки якості сигналу можуть відрізнятися в залежності від того, які дані використовуються:

- ЕКГ сигнал – підходи, що засновані на чистій ЕКГ;

– анотовані дані – підходи, що для обробки сигналу використовують анотацію ЕКГ сигналу (виявлення ключових точок кожного удару серця);
– ЕКГ сигнал та анотовані дані – підходи, що мають змішану модель перевірки якості сигналу.

3) *Математичний підхід*. При огляді літератури було виявлено два основних підходи, з точки зору математики, за допомогою яких проводиться аналіз:

– нейронні мережі – підходи, у яких створюється модель аналізу з використанням штучного інтелекту, на основі досить великої вибірки ЕКГ сигналів[7, 8];
– статистичний аналіз – підходи, що засновані на створенні алгоритмів подібних до дерев рішень, з використанням статистично визначених порогів (метрики що перевіряють якість сигналу, за значення пульсу, довжини *RR*-інтервалів) [9].

4. Метод перевірки якості ЕКГ на основі системи правил

Даний метод було розроблено для валідації якості сигналів електрокардіограми, що вимірюється за допомогою одноканального ЕКГ.

Метод заснований на 5 правилах перевірки якості сигналу.

Правило 1. Виключити всі записи, де не вистачало принаймні одного відведення. Якщо сума по всьому відведенню була менше 0, запис признається «неякісним».

Правило 2. Виконати пошук зміни амплітуди обмеженої тривалості, коли виникає спотворення сигналу, який, швидше за все, спричинений поганим контактом електрода.

Правило 3. Виконати пошук високого амплітудного піку, тобто абсолютної величини максимумів. (Правило 3 тісно пов'язане з Правилом 2.)

Правило 4. Перевірити, чи є на кардіограмах наявний дрейф ізолінії.

Правило 5. Виявити шумні сигнали. Використовуються порогові значення для стандартного відхилення, нормованого на амплітуді відведення.

5. Розроблений метод перевірки якості ЕКГ

Розроблений метод складається з двох частин:

– перевірка наявності контакту;
– валідація якості сигналу за допомогою згорткової нейронної мережі.

Відслідковування наявності контакту при вимірі сигналу електрокардіограми необхідно, для того, щоб уникнути ситуації, коли сигнал є валідним з точки зору автоматизованих методів (значення *RR*-інтервалів залишаються в нормі, пульс, або інші дані залишаються в нормі), але з точки зору реального ЕКГ сигналу, ми отримаємо змінений удар серця. На рис. 3 наведено приклад абсолютно якісного ЕКГ сигналу, та червоним виділено відрізок кардіограми, без якого новий сигнал все ще залишиться валідним, з точки зору валідації сигналу за допомогою автоматизованих методів:

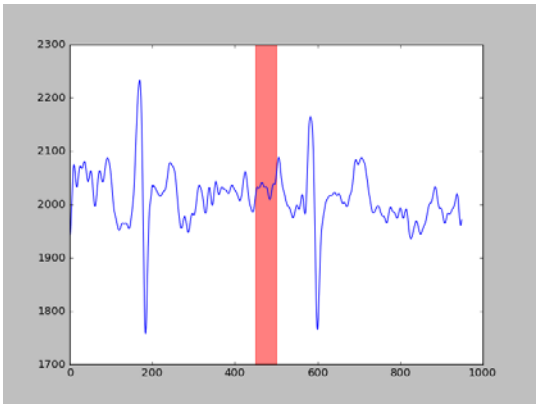


Рис. 3. Приклад сигналу, з імітацією відсутності контакту

Розроблена модель, що використовується для валідації ЕКГ сигналу являє собою згорткову нейронну мережу, що приймає на вхід 1000 точек кардіограми, а на виході видає ймовірність від 0 до 1 того, що поданий сигнал є «якісним». Дані перш ніж подаватись на валідацію нормалізуються.

Згорткова мережа складається з 6 згорткових шарів та 3 повнозв'язних шарів. У перших двох згорткових шарах 16 фільтрів, у других – 32. Між згортковими шарами знаходяться шари максимізаційного агрегування для

зменшення розмірності. Між згортковими та повнозв'язними шарами знаходиться дропаут шар для регуляризації. Функція активації у всіх шарах – ReLU.

Вчилась нейронна мережа 12 епох, алгоримом градієнтного спуску з кроком 0.003. Структура нейронної мережі підбиралась на основі результатів перевірки точностей, при зміні структури. Результати тестування для різної кількості згорткових шарів наведено в таблиці 1.

Табл. 1. Точності алгоритму для різної кількості згорткових шарів

Кількість шарів	Отримана точність(%)
3	85.0
4	87.7
5	92.3
6	95.9
7	95.0

6. Інтерпретація отриманих результатів

За результатами дослідження, було створено метод перевірки якості ЕКГ сигналу на основі нейронної мережі, та відслідковування наявності дотику до електродів, розробка алгоритму проводилась на 17000 семплів кардіограм, що були розмічені на два кластери: "якісні" та "неякісні". Перевірка роботи алгоритму проводилась на 4000 семплів, також розмічених за допомогою експертів. Для оцінки результатів роботи отриманого алгоритму, було проведено порівняння з алгоритмом перевірки якості ЕКГ сигналу, що діє на основі правил. У дослідженні, що описує даний алгоритм, було показано, що він має досить низьку точність, але його перевага в швидкості, тому цей алгоритм, можна вважати нижньою планкою точності, результати порівняння розробленого алгоритму (згорткова нейронна мережа) та алгоритм, що діє на основі правил на даних зібраних з одноканального кардіографа наводяться нижче (табл. 2).

Табл. 2. Порівняння точностей розробленого алгоритма, та описаного у літературі

	Точність класифікації неякісних кардіограм	Точність класифікації якісних кардіограм
Алгоритм що діє на основі правил	0.63	85.0
Розроблений алгоритм	0.931	0.965

Отримані результати підтверджують, що з точки зору задачі перевірки якості сигналу електрокардіограми, цей сигнал можливо розглядати, як сигнал з темпоральним патерном. Також, важливим елементом при перевірці якості ЕКГ сигналу, є відслідковування наявності контакту, з технічної точки зору, так як склеїний сигнал може визнаватися абсолютно валідним з точки зору математичних методів перевірки якості, але нестиме в собі некоректну інформацію (довжина отриманих *RR*-інтервалів може відрізнятися від реальної). Отримана точність для одноканального кардіографа, перевищує існуючі статистичні методи, які є більш чутливими до шумів.

7. Заключення

Досліджено класифікацію існуючих методів перевірки якості сигналу електрокардіограми. Розроблено метод перевірки якості електрокардіограми знятої з одноканального кардіографа на основі алгоритму класифікації за допомогою згорткової нейронної мережі та перевірки наявності контакту. Проведено аналіз отриманих результатів на основі порівняння розробленого методу та методу перевірки якості електрокардіограми, на основі вирішальних правил.

Список літератури

1. Damrong Sukitpunyaroj “ECG Interpretation: the basics”, Perfect Heart Institue, Piyavate Hospital
2. L. Wieclaw, Y. Khoma, P. Falat, D. Sabodashko, V. Herasymenko “Biometric Identification from Raw ECG Signal Using Deep Learning Techniques”, Lviv Polytechnic National University
3. Nitzan M. et al. The variability of the photoplethysmographic signal – a potential method for the evaluation of the autonomic nervous system. – Physiol. Means, 1998, v.19, p.93
4. “Heart rate variability Standards of measurement, physiological interpretation, and clinical use”/ guidelines. European Heart Journal (1996)
5. A. Dupree, ms, Sarah Vincent, ms, Paul A. Iaizzo, phd. “Basic ECG theory, recordings and interpretation”
6. Catalina Lionte, Cristina Bologa and Laurentiu Sorodoc. “Toxic and Drug-Induced Changes of the Electrocardiogram”. Catalina Lionte, Cristina Bologa and Laurentiu Sorodoc ”Gr.T.Popa” University of Medicine and Pharmacy, Iasi, Romania
7. Eduardo Morgado, Felipe Alonso-Atienza, Ricardo Santiago-Mozos, Óscar Barquero-Pérez, Ikaro Silva, Javier Ramos,corresponding author and Roger Mark. “Quality estimation of the electrocardiogram using cross-correlation among leads”. Biomed Eng Online. V. 14, 2015
8. Borisav Jovanović, Vančo Litovski, Milan Pavlović “QRS complex detection based ECG signal artefact discrimination”. Series: Electronics and Energetics Vol. 28, No 4, December 2015, pp. 571 – 584
9. Vaclav Chudacek, ˇLukas Zach, Jakub Kuzılek,, Jirı Spilka, Lenka Lhotska. “Simple Scoring System for ECG Quality Assessment on Android Platform”. Department of Cybernetics, Faculty of Electrical Engineering, Czech Technical University in Prague, Czech Republic

УДК 004.93(015.7)

РИБАЛЬЧЕНКО О. В.,
ГУЛЯНИЦЬКИЙ Л.Ф.

ОГЛЯД ТА РОЗВ'ЯЗУВАННЯ ЗАДАЧІ МАРШРУТИЗАЦІЇ БПЛА

Подано формальну постановку задачі маршрутизації безпілотних літальних апаратів, в якій розглядаються одночасно етапи планування операції та відвідування літальними апаратами встановлених цілей. Розроблено метаевристичний алгоритм розв'язування та проведено його аналіз на основі результатів аналізу обчислювального експерименту.

The formal statement of the particular UAV routing problem is covered. The problem under examination unites the stages of pre-mission planning and visiting targets. The methaeuristic algorithm for solving the problem have been developed and analyzed based on the results of computational experiment.

1. Вступ

Важливість використання безпілотних літальних апаратів (далі - БПЛА) у наш час важко переоцінити, особливо в умовах бойових дій. Наприклад, можливість проведення розвідки та відеозйомка об’єктів супротивника без ризику для людського життя надає значну перевагу при плануванні операцій. В рамках даної статті буде розглянуто ситуацію, за якої певна кількість БПЛА, що мають визначені характеристики, розміщується на різних базах та здійснюють вильоти для зйомки визначених територій. Кінцевою метою такої операції буде отримання знімків цих територій. Планування операції покладається на програмний комплекс, постановка задачі якого буде наведена далі. Задача планування операції надалі позначатиметься як UAVRP (unmanned aerial routing problem) [1]. Слід також зазначити, що програмний комплекс, хоч і призначений у першу чергу для розвідки, можна адаптувати без значних змін до багатьох інших галузей та практичних потреб. Наприклад:

- збирання інформації з автономних датчиків для моніторингу стану ґрунту чи рослин;
- відслідковування стану водоймищ;
- проведення пошукових операцій з повітря.

2. Опис задачі

Планувальник операції отримує наступну інформацію:

- карту місцевості проведення операції;
- набір координат цілей для зйомки;
- набір координат баз БПЛА, з яких можуть здійснюватися вильоти та які можуть приймати апарати;
- набір БПЛА та їх характеристик - дальність польоту, швидкість.

Цільова функція описується як знаходження такого плану польотів, щоб кожна ціль була відвідана, а сумарні витрати енергії були якомога меншими.

Зведемо цю задачу до добре відомої задачі маршрутизації транспорту (Vehicle Routing Problem, далі - VRP) [2].

3. Огляд класифікації задач VRP

Розподілення товарів є однією з ключових функцій у логістиці та включає їх доставку від виробника до споживача через транспортну мережу. Такі переміщення вимагають великих витрат, особливо для галузі доставки. VRP - загальна назва класу задач комбінаторної оптимізації, в яких споживачі обслуговуються певною кількістю машин. Машини базуються у депо, виїжджають для обслуговування споживачів (доставки), після чого повертаються. Існує велика кількість різноманітних варіантів постановок задачі - з обмеженнями на тривалість, час доставки, вантажопідйомність машин, максимальну довжину шляху тощо. Класична задача маршрутизації транспорту задається орієнтованим графом $G(E,V)$, де $V=\{0, 1, \dots, n\}$ задає набір транспортних вузлів (вершин), E - множину ребер. Депо (база) записується як вузол $j=0$, а споживачі $j = 1, 2, \dots, n$, кожен з яких характеризується потребою $d_j > 0$. Кожне ребро відповідає шляху з вершини i

до вершини *j*. Вага кожного ребра $C_{ij} > 0$ відповідає вартості (часу, відстані тощо) переїзду з *i* до *j*. Якщо $C_{ij} = C_{ji}$, задача є симетричною, інакше - асиметричною. З точки зору складності, VRP відноситься до класу NP-складних задач, оскільки узагальнює задачу комівояжера (далі - TSP) та задачу про пакування контейнерів (далі - BPP), що є добре відомими NP-складними задачам [3]. Математичне формулювання класичної VRP можна знайти у роботі [4]. Багато робіт присвячено як класичній версії VRP, так і її модифікаціям. Наведемо деякі з них:

- з обмеженням на вантажопідйомність (Capacitated VRP, далі – CVRP) – обсяг вантажу на кожному маршруті не повинен перевищувати граничного значення [5];
- з часовими вікнами (VRP with Time Windows, далі – VRPTW) – для відвідування вершин існує певний допустимий проміжок часу [6];
- з багатьма депо (Multiple Depot VRP, далі – MDVRP) – відправлення може здійснюватися з багатьох депо [7];
- з різноманітним транспортом (Split Delivery VRP, далі – SDVRP) – характеристики транспортних засобів можуть відрізнятися [8].

Частину особливостей цих модифікацій можна використати при описанні наведеної вище UAVRP у термінах VRP. Виліт літаків, як і їх посадка, може здійснюватися навіть з пересувної позиції посеред поля, тож оскільки точки початку і кінця польоту може не співпадати, а різні літаки можуть мати відмінні бази, можна скористатися елементами постановки MDVRP. Літаки можуть мати різні характеристики (зокрема, відстані польоту без дозаправки та радіус охоплення поверхні при зйомці) – що є ключовою особливістю SDVRP. Також через світлові та погодні умови має сенс обмеження на повернення літаків, що можна представити у вигляді часових вікон для пунктів повернення, тож наявні елементи VRPTW. Це дозволяє пов'язати елементи задачі з широковідомими модифікаціями VRP, а отже, використати основні їх підходи до побудови математичної моделі UAVRP.

4. Математична постановка задачі

Планувальник операції отримує наступну інформацію:

n – кількість пускових точок (баз БПЛА), з яких можуть здійснюватися вильоти та які можуть приймати апарати;

m – кількість БПЛА;

m_{j0} – максимальна дальність польоту j-го БПЛА, км;

m_{j1} – швидкість польоту j-го БПЛА, км/год;

k – кількість цілей, що необхідно відвідати;

C_{ij} – відстань між точками *i* та *j*.

Окремо зазначимо, що застосування формули обчислення відстані між точками на двовимірній площині призведе до похибки через викривлення траєкторії при переміщенні над поверхнею. Для розрахунку відстані по поверхні Землі використаємо формулу гаверсинуса – важливе рівняння у навігації, яке дозволяє обчислити відстань між точками на сфері, за їх довготою та широтою. Є окремим випадком більш загальної формули сферичної тригонометрії, закону гаверсинусів, відносно сторін та кутів сферичних трикутників [9]. Перша таблиця гаверсинусів була опублікована Джеймсом Ендрю у 1805 р.:

$$a = \sin^2\left(\frac{|\phi_1 - \phi_2|}{2}\right) + \cos\phi_1 \cos\phi_2 \sin^2\left(\frac{|\lambda_1 - \lambda_2|}{2}\right),$$

$$c = 2\arctan\left(\sqrt{\frac{a}{1-a}}\right),$$

$$d = Rc$$

де φ – широта, λ – довгота, *R* – радіус Землі (середній = 6,371 км). *d* – шукана відстань.

Висотою польоту при обчислюванні відстані буде знехтувано.

Виконання перельоту з точки *i* у точку *j* апаратом *l* описуватиметься змінними x_{ijl}:

$$x_{ijl} = \begin{cases} 0, \textit{ переліт не виконується} \\ 1, \textit{ переліт виконується} \end{cases},$$

$$i, j = \overline{1, n+k},\ l = \overline{1, m}. \qquad (1)$$

Загальна відстань *D*, яку мають подолати всі БПЛА під час виконання завдання згідно з отриманим планом операції обчислюється наступним чином:

$$D = \sum_{i=1}^{n+k} \sum_{j=1}^{n+k} \sum_{l=1}^{m} x_{ijl} C_{ij} \,. \qquad (2)$$

Загальний час виконання завдання T згідно з отриманим планом операції обчислюється згідно з формулою:

$$T = \max_{l=1,m} \frac{1}{m_{l1}} \sum_{i=1}^{n+k} \sum_{j=1}^{n+k} \sum_{l=1}^{m} x_{ijl} C_{ij} \,, \qquad (3)$$

де m_{l1} – згадана вище швидкість l-го БПЛА.

Довжина шляху кожного БПЛА не має перевищувати його максимально допустимої відстані перельоту:

$$\sum_{i=1}^{n+k} \sum_{j=1}^{n+k} x_{ijl} C_{ij} \le m_{l0} \,,\ l = \overline{1,m} \,. \qquad (4)$$

Кожен БПЛА відвідує певну кількість цілей, отже, прибуття та відправлення від кожної цілі має здійснитися лише один раз. Також, кожна ціль має бути відвідана строго одним БПЛА. Отже, для перевірки коректності розв’язку з точки зору обходу цілей, введемо наступні обмеження.

Один і тільки один БПЛА може покинути ціль:

$$\sum_{l=1}^{m} \sum_{i=1}^{n+k} x_{ijl} C_{ij} = 1 \,,\ j = \overline{n+1, n+k} \,. \qquad (5)$$

Один і тільки один БПЛА може прибути до цілі:

$$\sum_{l=1}^{m} \sum_{j=1}^{n+k} x_{ijl} C_{ij} = 1 \,,\ i = \overline{n+1, n+k} \,. \qquad (6)$$

Слід зазначити, що у формулах (5) та (6) при перевірці кількості перельотів, до цілі та від цілі відповідно, переглядаються також перельоти між депо та цілями. Проте перевірка за цими формулами для самих депо не відбувається.

Таким чином, виконується вимога відвідування кожної цілі. Наступна група обмежень пов’язана з вильотами з депо. Необхідно забезпечити, щоб кожен з БПЛА, що має відвідати набір цілей, виконав переліт від одного з депо до цілі, яка не має вхідних перельотів від інших цілей.

Означимо M як множину індексів апаратів, що мають призначені цілі у отриманому плані.

Умова вильоту БПЛА з депо:

$$\sum_{i=1}^{n} \sum_{j=n+1}^{n+k} x_{ijl} = 1 \,,\ l \in M \,. \qquad (7)$$

Умова повернення у депо:

$$\sum_{i=1}^{n} \sum_{j=n+1}^{n+k} x_{jil} = 1 \,,\ l \in M \,. \qquad (8)$$

Виконання даних умов забезпечує перевірку коректності маршруту кожного БПЛА. Якщо в отриманому плані виконання завдання наявні апарати без призначених цілей, обмеження (7) та (8) не мають сенсу, тому перевірка відбувається для індексів з множини M.

Для розв’язування сформульованої задачі (1)–(8) розроблено наближений алгоритм оптимізації мурашиною колонією (ОМК) з використанням детермінованого локального пошуку (ДЛП) в якості вбудованої процедури [10].

5. Дослідження ефективності розробленого алгоритму

Дослідження проводилося на основі аналізу результатів обчислювального експерименту із розв'язування задачі маршрутизації БПЛА з наступними параметрами:

- $n = 4$ – кількість депо;
- $m = 3$ – кількість БПЛА;
- $k = 23$ – кількість цілей.

Кожній цілі та депо відповідає точка на карті, програма отримує на вхід широту та довготу точки.в

Використовувалося таке апаратне забезпечення: процесор Intel Core i7, 2.2 GHz, з OSX та оперативною пам’яттю 16 GB.

В таблиці наведені результати розв'язування задачі з 23 цілями, 4 депо та 3 БПЛА з використанням алгоритму ОМК без двокрокових переходів (стандартний мурашиний алгоритм); алгоритму ОМК з двокроковими переходами (назвемо це ОМК з далекоглядними мурахами); алгоритму ДЛП, що базується на перестановках Ліна.

Для алгоритмів ОМК було здійснено 10 запусків з датчиками псевдовипадкових чисел з рівномірним розподілом, ініціалізованими різними початковими значеннями. Алгоритм ДЛП було запущено окремо 10 разів, у якості початкових значень використано результат, отриманий на першій ітерації кожного запуску ОМК з двокроковими переходами. У таблиці 1 наведено усереднені результати 10 взаємно незалежних між собою запусків для кожного розглянутого алгоритму.

Для обчислення відносної похибки ε використовується f^* – найменше відоме значення цільової функції для даної задачі ($f^* = 253,097$), а відносна похибка розв'язку обчислюється за формулою:

$$\varepsilon = 100 \cdot (f - f^*) / f^*.$$

З таблиці можна зробити висновок, що розроблений алгоритм з двокроковими переходами дозволив отримати істотно кращий середній результат.

Табл. 1. Усереднені результати обчислень

Алгоритм	**Класичний ОМК**	**ОМК з далекогляд-ними мурахами**	**ДЛП**
Цільова функція *f*, км	269,8836	255,4535	278,6548
Відносна похибка ε, %	6,63	0,93	10,09
Час, с	40,22	43,06	1,03

6. Висновки

Здійснено огляд варіацій задач маршрутизації БПЛА у різних предметних галузях та їх концептуальні постановки. Описано зведення задачі маршрутизації БПЛА до VRP. Наведено особливості окремих класів задач VRP, що можуть бути застосовані до задачі, що розглядається. Сформульовано та формалізовано один з варіантів задачі маршрутизації БПЛА, запропоновано і реалізовано алгоритм її розв'язування. Виконано дослідження розробленого алгоритму ОМК, отримані результати розв'язування задачі з реальними об'єктами на місцевості. Отримано відносну похибку результатів роботи кожного алгоритму, що дозволяє порівняти їх ефективність.

Список літератури

1. A. Alotaibi K. Unmanned Aerial Vehicle Routing In The Presence Of Threats / Kamil A. Alotaibi. – Arlington: : The University Of Texas At Arlington, 2014. P. 25-121.
2. Braekers K., Ramaekers K., Van Nieuwenhuyse I. The Vehicle Routing Problem: State of the Art Classification and Review // Computers & Industrial Engineering. – 2016. – 99. – P. 300-313.
3. Garey M. R. Computers and intractability: A guide to the theory of NP-completeness. / M. R. Garey, D. Johnson. – New York: W. H. Freeman and Company, 1979. – 338 с. – (4).
4. Laporte G. The vehicle routing problem: An overview of exact and approximate algorithms / Laporte G.. // European Journal of Operational Research. – 1992. – №59. – С. 345 – 358.
5. Baldacci R. Exact algorithms for routing problems under vehicle capacity constraints / R. Baldacci, P. Toth, D. Vigo. // Annals of Operations Research. – 2010. – №175. – С. 213 – 245.
6. Solomon M. Time Window Constrained Routing and Scheduling Problems / M. Solomon, J. Desrosiers. // Transportation Science. – 1988. – С. 1 – 13.
7. Renaud J. A tabu search heuristic for the multi-depot vehicle routing problem / J. Renaud, G. Laporte, F. Boctor. // Computers & Operations Research. – 1996. – С. 229 – 235.
8. Baldacci R. Routing a Heterogeneous Fleet of Vehicles / R. Baldacci, M. Battarra, D. Vigo. // The Vehicle Routing Problem: Latest Advances and New Challenges. – 2008. – С. 3 – 27
9. Van Brummelen G. Heavenly Mathematics: The Forgotten Art of Spherical Trigonometry. Princeton, N.J., Oxford: Princeton University Press, 2013. P. 160-162.
10. Гуляницький Л.Ф., Мулеса О.Ю. Прикладні методи комбінаторної оптимізації. К.: Видавничо-поліграфічний центр "Київський університет", 2016.

УДК 004.382:004.042

СУХАНЮК М.В.
ШИШКІН В.І.
СТЕЦЕНКО І.В.

ЕЛЕМЕНТИ МОДЕЛІ РОЗУМНОГО ВІДЕО-РЕЄСТРАТОРА

На сьогоднішній день великою популярністю користуються “розумні пристрої”. У даній статі розглянуто пристрій “розумний відео-реєстратор”, з можливістю розпізнавання українських авто номерів. Розглянуто основні принципи комунікації між серверами та пристроями. Описані основні модулі, з яких складається пристрій розумного відео-реєстратора, та з якою метою використовується кожний модуль. Представлена схема роботи системи в цілому.

Такий засіб зможе підвищити рівень захищеності на дорогах, що також є актуальним питанням.

Nowadays, "smart devices" are very popular. This article discusses the device "smart video registrar", with the ability to recognize Ukrainian automobile numbers. The basic principles of communication between servers and devices are considered. The main modules that make up the smart video recorder are described and for which purpose each module is used. The scheme of the system as a whole is presented.

Such a tool will be able to increase the level of security on the roads, which is also a topical issue.

Ключові слова: відео-реєстратор, розумні пристрої, ASUS Tinker Board, Satellite Based Augmentation System, GPS, SQLLite, NMEA, GPSD, JSON, бібліотеки для перетворення даних, Jackson, Protobuf, серіалізація даних, десереалізація даних, Apache ActiveMQ, кластер серверів, JMS.

1. Вступ

На сьогоднішній день великою популярністю користуються “розумні пристрої”. Такі пристрої зазвичай являються певною модернізацією вже звичних речей. У даній статі хочемо розглянути такий пристрій, як “розумний відео-реєстратор” з можливістю розпізнавання українських авто номерів.

Такий засіб зможе підвищити рівень захисту на дорогах, що також є актуальним питанням.

Далі розглядаються елементи моделі, які були використані при побудові моделі розумного відео-реєстратора.

2. Основа відео-реєстратору

Основою став одноплатний комп’ютер ASUS Tinker Board. Він випущений у 2017 році, має GPIO порти так само, як і Raspberry PI. Плата підтримує відео 4K, має 2 Гб оперативної пам'яті, гігабітний інтернет і процесор Rockchip RK3288 з частотою 1,8 ГГц [1]. Завдяки процесору Rockchip даний мікрокомп'ютер добре пристосований до обробки відео-інформації.

3. Опис компонентів

Складові компоненти розумного відео-реєстратора наведені далі.

1. GPS модуль від компанії WaveShare на основі NEO-6M. Даний модуль виконує функціональність, пов’язану з обчислення координат, швидкості, курсу і інших параметрів. Для знаходження відстані між приймачем і передавачем необхідно спочатку синхронізувати їх годинники, потім розрахувати шукану відстань, знаючи швидкість поширення радіохвилі та затримку між моментом передачі та моментом прийому, що виконується автоматично. Приймач використовується в системі для отримання координат транспортного засобу. NEO-6M має середню точність та невеликий розмір, що для є важливим аспектом. Neo-6M вміє використовувати супутникові системи диференціальної корекції SBAS (Satellite Based Augmentation System), що

збільшує точність визначення положення до 2 м, а також, AGPS (Assisted GPS) для зниження часу холодного старту.

2. Модуль камери від компанії Raspberry. Для роботи із Raspberry Pi Camera використовують інфрачервоні (NoIR) та звичайні камери (v2). В даному проекті використовується звичайна камера. Версія v2 має значні переваги над першою, надає можливість знімати FULL HD відео з кадровою частотою 30 fps та має кращу якість зйомки. Крім того, кут огляду у другій версії збільшився, а кількість мегапікселів збільшилась на 3 одиниці.

3. Монітор руху 10 DOF IMU Sensor (C) - модуль десяти ступенів свободи від Waveshare.

Даний модуль об'єднує в собі трьохосьовий гіроскоп, трьохосьовий акселерометр і трьохосьовий компас (магнітометр) на одному кристалі MPU9255 та барометричний датчик тиску BMP280. Вибір саме цього датчику обумовлений ціною та якістю зборки. Крім того, наявна можливість додатково доповнювати функціонал відео-реєстратору наступними можливостями: стабілізація камери, контроль нахилу відео-реєстратору і т.п.

4. Модем LTE. Якість інтернету грає вирішальну роль, оскільки використання розумного відео-реєстратора тісно пов’язане з постійним обміном даних між сервером та реєстратором. Через постійний рух автомобіля, на якому встановлений відео-реєстратор, можуть спостерігатися збої у передачі даних або мережеві ями.

5. СУБД SQLLite. Вибір даної СУБД обумовлений наступними факторами:

- невеликий розмір,
- популярність серед портативних пристроїв,
- швидкість,
- надійність.

Крім того, СУБД SQLLite підтримує SharedPreferences - це сховище ключів / значень, в якому зберігаються дані під певним ключем, а саме, координати транспортного засобу, на якому встановлений розумний відео-реєстратор.

На рисунку 1 представлені основні модулі відео-реєстратору, їхні складові, та як вони взаємодіють один з одним.

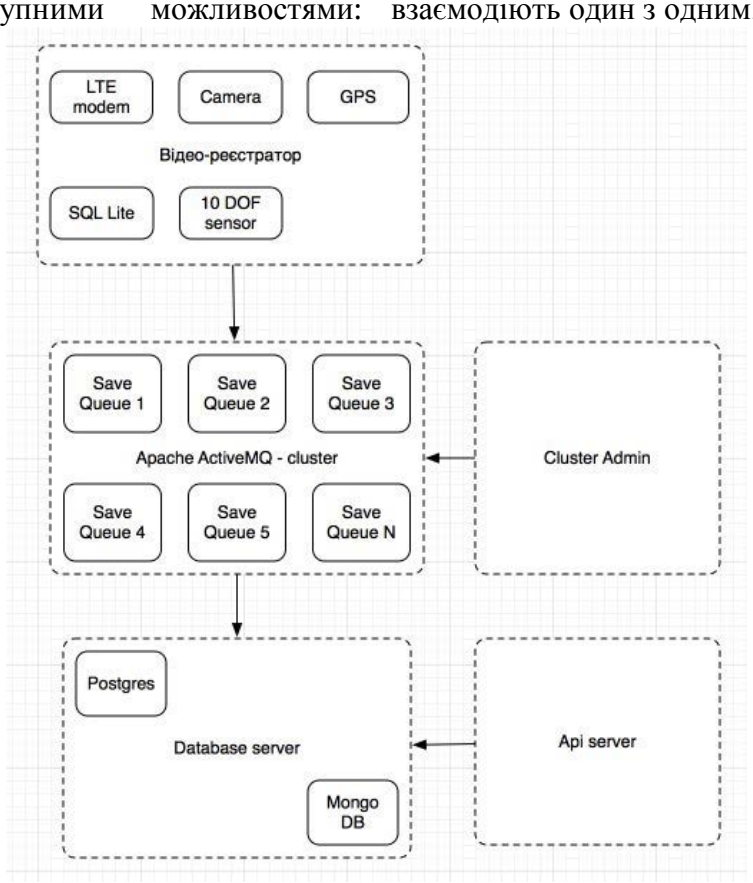


Рис. 1. Схема ключових модулів проекту

4. Підключення датчику

До Asus Tinker Board був підключений датчик NEO-6M, за допомогою якого відслідковується положення транспортного засобу. Даний пристрій передає дані, які він отримав від супутників, у форматі NMEA кожні N секунд. Дуже важливою є інформація про ширину, довготу, висоту та швидкість руху об'єкта. Для підключення датчика до Asus tinker board необхідно з'єднати контакти плати та датчик NEO-6M за наступною схемою, що наведена у таблиці 1.

Табл. 1. Підключення датчику до Asus Tinker Board

Asus Tinker Board/ Raspberry PI	**Waveshare GPS Neo6m**
Pin 1 (3.3 V)	VCC
Pin 8 (TX)	RX
Pin 10 (RX)	TX
Pin 6 (GND)	GND

5. Робота з форматом NMEA

Для роботи з датчиком NEO-6M використовується бібліотека GPSD [2]. Дана бібліотека підтримує виконання таких функції:

1) синтаксичний розбір вхідного потоку даних у форматі NMEA з датчику GPS Neo6m;
2) перетворення даних у формат JSON;
3) фільтрування даних від непотрібної або неправильної інформації;
4) надає можливість передавати дані за допомогою TCP/UPD та інших популярних протоколів.

При роботі з форматом NMEA необхідно враховувати його команди. Наведемо перелік основних команд NMEA 0183 версії 3:

- GPGGA – дані про останнє місцезнаходження;
- GPGLL – координати широта/довгота;
- GPGSA – DOP(GPS) активні супутники;
- GPGSV – спостережувані супутники;
- GPWPL – параметри заданої точки;
- GPBOD – азимут однієї точки відносно іншої;
- GPRMB – рекомендований мінімум навігаційних даних для досягнення заданої точки;
- GPRMC – рекомендований мінімум навігаційних даних;
- GPRTE – маршрути;
- HCHDG – дані від компасу [3].

Бібліотека GPSD має певні недоліки та не відповідає всім вимогам проекту, який розробляється. За результатами проведеного тестування навантаження дана бібліотека не є стабільною, якщо встановлена велика кількість з'єднань. Використання TCP/UDP з'єднання не є доречним в цьому випадку, оскільки при пересуванні транспортного засобу інтернет має властивість пропадати, що призводить до постійного оновлення підключення. Дані знайдених транспортних засобів та положення транспортного засобу передаються одним блоком даних.

Бібліотека GPSD написана на мові C. Для того, щоб інтегрувати її в проект, був написаний клієнт на Java, який підключається до відкритого серверу GPSD (є доступним тільки локально), обмінюється інформацію та передає необхідні команди.

6. Перетворення даних

Бібліотеки GPSD надає дані у форматі JSON, проте для подальшої роботи необхідно перетворити ці дані у більш зручний для обробки формат. Для забезпечення ефективності обробки даних необхідна висока швидкість серіалізації та десеріалізації та мінімально можливі витрати пам'яті [4]. Було проведено тестування найпопулярніших бібліотек для мови програмування Java GSON [5] та Jackson [6]. Бібліотека GSON виявилась швидшою у результаті випробування, проте її швидкість не достатньо для використання в проекті.

Через низьку ефективність формату JSON, для передачі даних між сервером та пристроєм в даному проекті використовується бібліотека ProtoBuf від компанії Google [7], яка є бінарною та за результатами тестувань є швидшою за GSON.

Далі наведено результати тестування бібліотек GSON, Jackson и Protobuf за такими критеріями:

1) швидкість серіалізації і десеріалізації для 100000 ітерацій;

2) швидкість серіалізації і десеріалізації для 10000000 ітерацій;
3) використання CPU;
4) динаміка використання пам'яті.

Результати випробувань наведені у таблиці 2.

Табл. 2. Результати випробувань

Критерій	GSON	Jackson	Protobuf
Швидкість серіалізації і десеріалізації для 100000 ітерацій, мс	445	13544	210
Швидкість серіалізації і десеріалізації для 10000000 ітерацій, мс	3223	91877	903
Використання ЦП	12,5%	12,6%	12,1%
Розмір задіяних ресурсів на останній ітерації, Б	39 088 584	59 816 548	49 696 136

7. Apache ActiveMQ

Для обміну даними між відео-реєстратором та сервером в якості брокера повідомлень використовується технологія, розроблена компанією Apache під назвою ActiveMQ. Дана технологія реалізує сервіс Java Message Service 1.1 (JMS), що може бути застосований окрім Java на інших мовах програмування, таких як Python, C++, .NET та ін. Сервіс підтримує наступні можливості:

- кластеризация;
- зберігання повідомлень;
- можливість використання різних баз даних;
- кешування;
- ведення журналів.

Для реалізації серверної частини використовується кластер з N серверів Apache ActiveMQ. Це надає можливість розвантажити кластер, який не здатний обробити вхідний потік інформації з усіх пристроїв. За рахунок використовуваної розподіленої технології існує можливість додавати сервери до кластеру для розвантаження та збалансування черги запису до бази даних.

8. Висновок

Розглянуті основні елементи для розробки інформаційної технології розумного відео-реєстратора: ASUS Tinker Board, датчики, перетворення даних, клієнтський сервер.

Обраний оптимальний формат даних, який забезпечує прийнятну швидкість передачі та прийому. За результатами тестування доведено, що Protobuf має більш високі показники якості в порівнянні з бібліотеками GSON та Jackson.

Наведена архітектура моделі розумного відео-реєстратора із описом компонентів.

Список літератури

1. ASUS Tinker Board [Електронний ресурс] – Режим доступу до ресурсу: https://tinkerboarding.co.uk/wiki/index.php?title=Hardware.
2. El-Rabbany A. Introduction to GPS: The Global Positioning System / Ahmed El-Rabbany., 2002. – 176 с.
3. NMEA data [Електронний ресурс] – Режим доступу до ресурсу: http://www.gpsinformation.org/dale/nmea.htm.
4. Kleppmann M. Designing Data-Intensive Applications / Martin Kleppfmann. – Sebastopol: O'Reilly Media, 2010.
5. Class Gson [Електронний ресурс] – Режим доступу до ресурсу: https://google.github.io/gson/apidocs/com/google/gson/Gson.html.
6. Jackson 2 – Convert Java Object to / from JSON [Електронний ресурс] – Режим доступу до ресурсу: https://www.mkyong.com/java/jackson-2-convert-java-object-to-from-json/.
7. Protocol buffers [Електронний ресурс] – Режим доступу до ресурсу: https://developers.google.com/protocol-buffers/.

УДК 004.021

ДАНИЛЬЧУК Р.К.

ЗАДАЧА КЛАСТЕРИЗАЦІЇ АДРЕС В МЕРЕЖІ БЛОКЧЕЙН

У даній статті розглянуто практичне застосування методу кластеризації адрес в мережі блокчейн на прикладі задачі визначення кількох адрес одного користувача. Результати дослідження показують доцільність пропонованого підходу до кластеризації адрес Bitcoin. Користувачам може бути корисно уникнути небезпечних моделей використання Bitcoin, а дослідникам провести більш розширений аналіз анонімності.

In this article, the practical application of the method of clustering of addresses in the blockade network is considered on the example of the task of determining the multiple addresses of one user. The results of the study show the appropriateness of the proposed approach to clustering Bitcoin addresses. It may be useful for users to avoid dangerous patterns of Bitcoin use, and for researchers to conduct a more advanced analysis of anonymity.

1. Вступ

Blockchain (з англ. block - блок, chain - ланцюг) - це ланцюжок блоків транзакцій, які зберігаються на комп'ютерах учасників ланцюжка. Кожен наступний блок пов'язаний з попереднім і складається з набору записів. Нові блоки завжди додаються лише в кінець цього ланцюжка [1].

Ланцюжок даних має три основні принципи:

- захищеність;
- розподіленість;
- відкритість;

Всі учасники блокчейну об'єднуються в комп'ютерну мережу. На кожному сервері зберігається копія всіх даних блоку. Це і є основою надійності blockchain.

Адже, щоб зламати ланцюжок, потрібно отримати доступ до бази даних всіх комп'ютерів мережі.

Всі дані, що з'являються в блоках відкриті (користувачі бачать їх) і зашифровані (користувачі не знають, кому вони належать).

Приклад запису в мережі блокчейн: "Користувач з ключем К отримав у кредит телефон з ключем S".

Кожен користувач може мати декілька різних ключів. Тобто, навіть знаючи ключ власника телефону, не можна дізнатися про наявність у нього штрафу за порушення правил дорожнього руху.

2. Підходи до вирішення задачі кластеризації

На рисунку 1 представлена класифікація алгоритмів та методів кластерного аналізу.

Сутність ієрархічних агломеративних методів полягає у тому, що на першому кроці кожний об’єкт вибірки розглядається як окремий кластер. Процес об’єднання кластерів відбувається послідовно, на підставі матриці відстаней або матриці подібності поєднуються найбільш близькі об’єкти. Послідовність об’єднання легко піддається геометричній інтерпретації й може бути представлена у вигляді графа-дерева. Основною передумовою ієрархічних дивізивних методів є те, що спочатку всі об’єкти належать до одного кластеру. У процесі класифікації за певними правилами поступово від цього кластера відокремлюються групи схожих між собою об’єктів [2]. Так, на кожному кроці кількість кластерів зростає, а міра відстані між кластерами зменшується. Складнощі ієрархічних методів кластеризації наступні:

- обмеження обсягу набору даних;
- вибір міри близькості;
- негнучкість отриманих класифікацій.

Перевага цієї групи методів порівняно з неієрархічними методами полягає у їх наочності і можливості отримання детального уявлення про структуру даних. При використанні ієрархічних методів

існує можливість досить легко ідентифікувати викиди в наборі даних і в результаті підвищити якість даних. Велика кількість методів ієрархічного кластерного аналізу різниться не тільки використаними мірами подібності (розходження),але й алгоритмами класифікації.

Неієрархічні методи виявляють більш високу стійкість по відношенню до викидів, невірного вибору метрики, включення незначущих змінних в базу для кластеризації та інше. Необхідно заздалегідь фіксувати результуючу кількість кластерів, правило зупинки і, якщо на те є підстави, початковий центр кластеру, що суттєво впливає на ефективність роботи алгоритму. Якщо немає підстав штучно задавати ці умови, рекомендується використовувати ієрархічні методи.

Для аналізу транзакцій, окрема транзакція розглядається як упорядкована $t = (A, B, c)$ та складається з:

- кінцевого багатоступеневого транзакційного входу A, де кожен вхід $(a_i, A_i) \in A$ – упорядкована пара адреси A_i і значення вхідного $a_i > 0$.
- кінцевого багатоступеневого транзакційного виходу B, де кожен вихід $(b_j, B_j) \in B$ – це упорядкована пара адреси B_j і значення вихідного $b_j \geq 0$.
- плати за транзакцію $c = \sum_{(a_i,) \in A} a_i - \sum_{(b_j,) \in B} b_j \geq 0$

Для довільної множини транзакційних входів або виходів A позначаємо мультимережний адрес в A, як Addr(A).

Найбільш очевидною ідеєю для

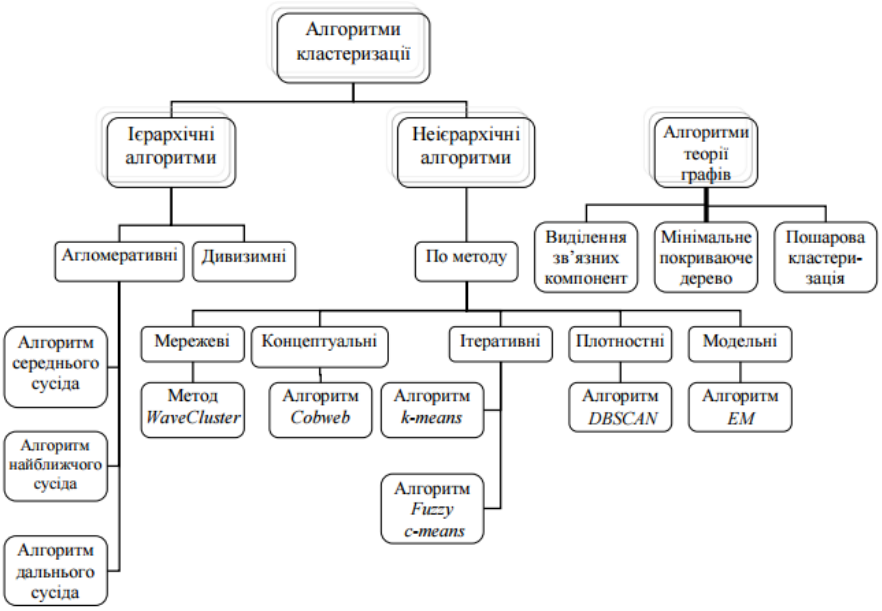


Рис. 1. Класифікація алгоритмів та методів кластерного аналізу

3. Алгоритм кластеризації адрес

Розглянемо мережі на основі блоків, які допомагають об’єднати групи адрес блокчейн в одну суцільну систему. Ці показники засновані на певних моделях, які є загальними для багатьох транзакцій в мережі. Однак вони не завжди задовольняються для всіх транзакцій, а отже схильні до помилок. Це означає, що деякі адреси можуть бути помилково пов'язані між собою.

кластеризації адрес блокчейну є з'єднання всіх вхідних адрес однієї транзакції. Якщо дві або більше адрес є входами однієї транзакції з одним виходом, то всі ці адреси керуються тим самим користувачем [3].

Розглянемо транзакцію t = (A, B, c), що задовольняє умови одноразової зміни.

- #Addr (B) = 2, тобто транзакція t має рівно два виходи.
- #Addr(A) = 6 ≠ 2, тобто кількість входів t не є рівною двом. Якщо #Addr(A) = #Addr(B) = 2,

транзакція, швидше за все, поділиться міткою передачі.

- Обидва виходи транзакції t, B_1 та B_2 не є обмінними адресами, тобто $B_1, B_2 \notin \text{Addr}(A)$.
- Один вихід транзакції B_1 не існував до транзакції t, а десяткове подання значення b_1 має більше ніж 4 цифри після крапки.
- Інший вихід транзакції B_2 раніше був частиною мережі, і в попередніх транзакціях він не був адресований поза обліковим записом.

Розглянемо алгоритм кластеризації адрес на прикладі мережі Bitcoin, який регулює баланс інформації, що надходить безпосередньо з блоків Bitcoin (CS та OTC) та додаткову інформацію, зібрану з Інтернету у вигляді тегів.

Нехай $T = \{t_j\}$ і є набором всіх транзакцій в блоці біткойн, тоді як A є набором всіх адрес, що присутні в транзакції з T.

Кластеризація адрес Bitcoin – це розбивка $A = A_1 \cup A_2 \cup \ldots \cup A_N$ на непересічні підмножини $A_l \cap A_l = \emptyset$ для $l \neq$ j. За допомогою $T_H \subset$ T позначимо сукупність всіх транзакцій, які задовольняють CS, або OTC. Для транзакції $t \in T_H$, через $\text{Addr}_H(t)$ позначимо множину всіх адрес, які слід віднести до одного користувача відповідно.

Інформація про теги представлена як сукупність негативних пар L = $\{(a_i, a_j,)\}$. Пара адрес $(a_i, a_j) \in$ L, якщо у нас є частина інформації про те, що ці адреси не контролюються одним і тим самим користувачем.

Слід зазначити, що як CS і OTC, так і позабіржова евристика і набір негативних пар L можуть містити помилкову інформацію.

Розглянемо різні типи спостережень:

- у випадку, якщо всі адреси Addr_H (t) для деяких t $\in T_H$ дійсно належать одному і тому ж користувачу з ймовірністю p;
- у випадку, якщо дві адреси $(a_i, a_j) \in L$ контролюються тим самим користувачем з ймовірністю q.

В інших випадках інформація про негативне об'єднання між будь-якою парою адрес в L перевіряється шляхом 1 - q.

Нехай ймовірність $P(A, T_H, L|p, q)$ буде функцією від кластеризації A, транзакції T_H та негативних пар L:

$$P(A, T_H, L|p, q) = \prod_{t \in T_H} p^{\mathbb{I}(\text{Addr}_H(t) \subset Cl(A))} \times (1-p)^{\mathbb{I}(\text{Addr}_H(t) \not\subset Cl(A))} \times \prod_{\{a,a,\} \in L} (1-q)^{\mathbb{I}(\{a,a,\} \not\subset Cl(A))} \times q^{\mathbb{I}(\{a,a,\} \subset Cl(A))},$$

де для деякого набору Bitcoin адреси S позначення $S \subset Cl(A)$ означає, що існує кластер A_l, такий, що $S \subseteq A_l$.

Отже, log-правдоподібність співвідноситься як

$$\ln P(A, T_H, L|p, q) = \sum_{T \in T_H} \mathbb{I}\big(\text{Addr}_H(t) \subset Cl(A)\big) \ln(1-p) + \sum_{\{a,a,\} \in L} \mathbb{I}(\{a, a'\} \subset Cl(A)) \ln(p) +$$

$$+ \sum_{T \in T_H} \mathbb{I}\big(\text{Addr}_H(t) \not\subset Cl(A)\big) \ln(p) + \sum_{\{a,a,\} \in L} \mathbb{I}\big(\{a, a'\} \not\subset Cl(A)\big) \ln(1-p)$$

Слід зазначити, що запропонована модель не призначена для використання імовірнісної структури реального світу, а лише дає більш розгорнутий підхід до систематичного вивчення довіри між різними джерелами інформації. Більше того, це дозволяє ефективно оптимізувати параметри.

Максимізація log-правдоподібності – це задача дискретної оптимізації, яка фактично NP-повна.

Розглянемо ретроспективно всі транзакції в мережі Bitcoin, які задовольняють одну евристику. На кожному етапі вирішується, чи приєднуються кластери, що відповідають адресі $\text{Addr}_H(t_J)$ до розглянутої транзакції t_J.

Нехай $A_J = A_{k1} \cup \ldots \cup A_{km}$ – об'єднання всіх кластерів, представники яких належать $\mathrm{Addr}_H\ (t_J)$.

Знайдемо зміни кількості негативних пар, що відповідають $\mathrm{Addr}_H\ (t_J)$ в один кластер A_J:

$$\Delta_{t_j}\left(\sum_{\{a,a,\}\in \mathrm{L}} \mathbb{I}\left(\{a,a'\} \not\subset Cl(A)\right)\right) = \sum_{\{a,a'\}\in \hat{A}_J} \mathbb{I}(\{a,a,\} \in A_l) - \sum_{i=1}^{m_j} \sum_{\{a,a'\}\in A_{k_i}} \mathbb{I}\left(\{a_i,a_j,\} \in A_l\right) = \Delta_{\hat{A}_j} - \sum_{i=1}^{m_j} \Delta_{A_j}$$

де ΔA_m - це кількість негативних пар в кластері A_m.

Тепер об'єднаємо всі кластери, що відповідають $\mathrm{Addr}_H\ (t_J)$, а отже зміна log-правдоподібності дорівнює

$$\Delta_P\left(t_j, A, L\middle|p,q\right) = ln\left(\frac{p}{1-p}\right) + \left(\Delta_{\hat{A}_j} - \sum_{i=1}^{m_j} \Delta_{A_j}\right) \ln\left(\frac{q}{1-q}\right)$$

Таким чином, якщо $\Delta_P\left(t_j, A, L\middle|p,q\right)$ є позитивним, то ми зливаємо всі кластери, що відповідають $\mathrm{Addr}_H\ (t_J)$, в іншому випадку потрібно продовжувати наступну транзакцію.

Слід відзначити, що завдяки такому підходу зміна параметрів p і q може призвести до дуже немонотонної зміни кластеризації. Наприклад, можна зменшити параметр q, який повинен вести до менших кластерів, але з'ясується, що найбільший кластер стає ще більшим.

4. Висновки

У цій роботі було проаналізовано існуючі методи для розв’язку задачі кластеризації та запропоновано використати алгоритм групування адрес блокчейн для визначення множини адресів одного користувача. В роботі наведений даний алгоритм. Та проаналізовано його особливості: використання для кластеризації не тільки інформацію про блокчейни, а й інформацію з Інтернету поза мережі блокчейн, та розгляд деяких типів даних поза мережею як голоси проти адресного об’єднання в процесі кластеризації. Такий підхід дозволяє уникнути значної частини помилкових об'єднань кластерів.

Список літератури

1. S. Nakamoto. (2008). Bitcoin: A peer-to-peer electronic cash system.
2. НФОРМАЦІЙНІ ТЕХНОЛОГІЇ УДК 004,825 к.т.н. Волосюк Ю.В. (ЄУ, м. Миколаїв) АНАЛІЗ АЛГОРИТМІВ КЛАСТЕРИЗАЦІЇ ДЛЯ ЗАДАЧ ІНТЕЛЕКТУАЛЬНОГО АНАЛІЗУ ДАНИХ
3. D. Ron and A. Shamir (2012). Quantitative analysis of the full bitcoin transaction graph. Cryptology ePrint Archive, Report 2012/584.

УДК 004.852

ЯВДОЩУК А. Р

АНАЛІЗ ПРОГРАМ ПІДТРИМКИ СТАРТАПІВ

У даній статті розглянуті платформи та програми підтримки стартапів. Визначені переваги та перспективи веб-застосування, що розробляється

Ключові слова: стартап, веб-застосування

This article discusses platforms and startup support programs. Defined advantages and prospects of web-application being developed

Keywords: startup, web application.

1 Вступ

Багато людей мріють створити свій стартап, маючи гарні ідеї для втілення, приєднатись до розробки вже існуючого, маючи необхідні вміння та навики, або фінансувати в проект, маючи достатньо коштів. Та сьогодні немає жодних платформ з пошуку команди для стартапу та підтримки її комунікації. Тому актуальним є створенні програмної платформи підтримки старап-команд, виконання старап-проектів та презентації їх інвестору.

.

2 Постановка проблеми

Стартап може розроблятися в будь-якому сегменті ринку, але частіше за все нові унікальні проекти створюються в області комп'ютерних та інноваційних технологій. Придумати і розвивати ідею в Інтернеті простіше і швидше, тому сучасні стартапи найчастіше створюються в IT-сфере. Однією з причин створення стартапу, його подальшого зростання на ринку і успішного просування є повільність великих корпорацій, які дохідність своїх продуктів отримують у вигляді виручки і не поспішають розробляти і впроваджувати нові ідеї і технології в своє виробництво. Тому саме стратап, будучи мобільними і рухливими, що є перспективною течією для будь-якої галузі ринку, становлять більшу конкуренцію великим корпораціям.

3 Мета статті

Метою статті є огляд схожих платформ та визначення унікальності та цінності проекту, який розробляється як платформа підтримки стартап-проектів, на ринку програмних продуктів.

4 Аналоги веб-сервісу підтримки стартапів

На даний момент людям дуже важко створювати проекти, адже пошук команди, комунікації між учасниками, управління проектами, слідкування за виконаними задачами та тими, які виконуються, відбувається на різних сервісах.

Наявні наступні програми: MS Project, Jira, Trac, Red Main, LinkedIN. Microsoft Project – програма управління проектами, розроблена компанією Microsoft. Jira – система відстеження помилок, призначена для організації спілкування з користувачами, в деяких випадках може бути використана для управління проектами. Програмне забезпечення дозволяє виявляти і класифікувати неполадки, розподіляти завдання і відстежувати роботу команди. Trac – вільне веб-застосування для управління проектами. Red Main – вільне серверне веб-застосування для управління проектами та відстежування помилок. До системи входить календар-планувальник та діаграми Ганта для візуального представлення ходу робіт за проектом та термінів виконання. LinkedIn, Фейсбук — соціальні мережі для пошуку і встановлення ділових контактів.

Головним конкурентом платформи, що розробляється, є застосування TeamFinding, за допомогою якого можна виконувати пошук команди, фахівців і однодумців та цікаві проекти. В Інтернеті існує велика кількість сайтів, на яких можна знайти інвестора для фінансування свого проекту.

Після об’єднання розробників платформи в команду стає питання про можливі сервіси для спілкування між собою та для управління проектом і виконанням задач. Для комунікацій застосовуються програми Skype, Slack, Telegram, What’sUp, залежно від того, чи необхідна відео-зв’язок, а для управління командою можна використовувати Trello, Kaiten, Planner.

Веб-застосування, що розробляється, буде об’єднувати в собі усі сервіси, які були перераховані вище: інструменти для зручного створення стартап-проектів, простий пошук команди та інвестора, комфортне спілкування з учасниками через аудіо- та відео- конференції, онлайн чат, дизайнерську дошку задач з відстеженням часу.

5 Переваги застосування, що розробляється

Веб-сервіс дозволятиме знаходити кандидатів у команду, які знатимуть та вмітимуть реалізовувати найрізноманітніші програмні інструменти. Керівник стартапу зможе запросити їх у свою команду, або вони самі відправлять заявку на виставлену вакансію. На зразок цього будуть проводитись пошуки інвестора: команда чи керівник буде знаходити фінансиста зі списку зареєстрованих на сайті, або сам інвестор зв’язується з керівником, якщо його зацікавила тема проекту.

В інтегрованому чаті можна буде з’ясовувати всі деталі договору.

Після набору команди будуть розподілятися існуючі задачі між усіма учасниками команди, а на дошці з завданнями буде виставлена уся необхідна інформація. Задачі можна буде фільтрувати за автором, виконавцем та тегам, над кожною дошкою будуть розташовані іконки виконавців та при клацанні на іконку показуватимуться задачі певного учасника.

Так як для найбільшої відповідності вимогам необхідно команді часто обговорювати усі деталі проекту, то буде в системі можливість проводити відео-конференції.

Описані основні аспекти системи підтримки стартап-проектів, будуть зручними та простими у використанні.

6 Інтеграція програмного забезпечення

По завершенні реалізації проекту підтримки стартапів, він інтегрується у мережу Інтернет. Автори розраховуються, що система швидко набере клієнтську базу і стане корисною багатьом користувачам через наявність людей з бажанням створювати стартапи. Система буде підтримуватися задля збереження користувачів та залучення нових як типова CRM-система. У планах на майбутнє – локалізація англомовна для надання можливості розробникам з різних країн об’єднуватись у команди для створення проектів.

Висновок

Проаналізувавши ринок програмного забезпечення, можна зробити висновок, що платформа, що реалізується, є унікальною та матиме достатньо переваг перед існуючими системами, адже проект матиме всі інструменти, завдяки яким можна легко створювати власні проекти. Даний продукт об’єднує ідеї та бізнес-задачі інших продуктів, але матиме особливості при плануванні часу, пошуку виконавців та інвесторів, а також організації задач і документів. Завдяки веб-сервісу можна буде з легкістю перетворювати ідеї та вміння у програмний продукт.

Список літератури

1. TEAMFINDING [Електронний ресурс] // Режим доступу: http://teamfinding.com/
2. Trello [Електронний ресурс] // Режим доступу:https://trello.com/
3. 5 онлайн-платформ для привлечения инвестиций [Електронний : ресурс] // Режим доступу: http://www.the-village.ru/village/business/cloud/150253-online-invest

УДК 004.021

ГЛЯНЬКО А.С.,
БАКЛАН І.В.

АВТОМАТИЗАЦІЯ БІЗНЕС-ПРОЦЕСІВ ДІЯЛЬНОСТІ ЛОГІСТИЧНОЇ КОМПАНІЇ ЗА ДОПОМОГОЮ WEB-ЗАСТОСУНКУ

В статті сформульовано математичну постановку задачі комівояжера в умовах вантажних перевезень з додатковими обмеженнями та запропоновано алгоритм розв‘язання цієї задачі. Серед додаткових умов наявні обмеження на максимальну можливу відстань, яку може подолати вантажівка без дозарядки.

This article formulated a mathematical statement of the task of the traveler in conditions of freight traffic with additional restrictions and proposed an algorithm for solving this problem. Among the additional conditions there are restrictions on the maximum possible distance that the truck can overcome without recharging.

1. Вступ

Для успішного ведення бізнесу потрібні вчасні логістичні перевезення різних груп товарів. Загальною проблемою в сучасних перевезеннях досі є проблема комівояжера. Водії вантажівок зазвичай користуються навігаторами або класичними мапами. Наразі з’являються нові електро-вантажівки, які дозволяють набагато зменшити вартість перевезень. Електро-вантажівки значно зменшать витрати на паливо та обслуговування вантажівок, а логістичні фірми збільшать свої прибутки. Але такі вантажівки мають набагато менший максимальний запас ходу. Маршрути, які будують стандартні навігатори не завжди будуть відповідати вимогам водіїв електричних вантажівок.

2. Постановка проблеми

Для того щоб не втрачати клієнтів, підприємцям завжди треба йти в ногу з часом. Не так давно Ілон Маск презентував світові нові електричні вантажівки. Ці вантажівки повинні замінити дизельні та бензинові вантажівки у найближче десятиріччя. Але зараз вони не дуже популярні. Основною проблемою сучасних електричних автомобілей є нерозповсюдженість АЗК, де наявні зарядки для таких авто. Водіям вантажівок, які долають тисячі кілометрів, необхідно бути впевненими, що вони не зупиняться посеред маршруту та не встигнуть в терміни доставки вантажу.

3. Мета статті

Метою даної статті є привернення уваги власників логістичних компаній до сучасних новинок автомобілебудування, а також демонстрація способу побудови коректних маршрутів для водіїв вантажівок при умовах деяких обмежень.

4. Основна частина.

Розглянуто задачу планування маршруту, де відомі бажані міста відвідування і максимальний запас ходу електро-вантажівки. Так як між двома містами може існувати декілька маршрутів, то у графі можуть бути наявними більше одного ребра між двома вершинами. Така задача є узагальненим випадком класичної задачі. Час заряджання електро-вантажівки постійний, вартість перевезення залежить від кількості товару, параметри вантажівки відомі. Розглянемо математичну постановку задачі комівояжера. Дано зважении орієнтовании мультиграф $G = (V, F)$, де $V = \{v_1, v_2, ..., v_n\}$ – множина вершин, які являють собою n міст; F – множина дуг, які відображають наявні маршрути між містами. Між двома вершинами може існувати декілька дуг, що відображає наявність декількох маршрутів між містами(доріг), які можуть відрізнятися за довжиною маршруту і часом знаходження у дорозі. Планування маршруту по містам має вкладатися в часові межі. Позначимо за F множину маршрутів $(v_i v_j)^t$ з міста в місто в

момент часу t, а f^t – конкретну дугу, що відповідає маршруту. Нехай ціна перевезення задається функцією cost (f^t)– невід'ємна залежна від часу функція, яка визначає загальну вартість перевезення товару з міста в місто j у час t. Необхідно знайти оптимальний за вартістю шлях з початкової вершини $s \in V$, що охоплює всі бажані міста та повертається знову у початкову вершину у заданийпроміжок часу $[t_{min}, t_{max}]$ чи повідомити що неможливо побудувати маршрут через задані міста.

Функція часу роботи автомобіля, що залежить від швидкості руху, кількості зупинок, простою і т.і.

$T_{max,min} = F(v_{руху}, n_{зупинок}, T_{простою})$**.** Метою задачі є будування таких маршрутів, що задовільняють усім умовам та на побудованному маршруті не існує такої ділянки без заправок з електричними зарядками, довжина якої перевищує максимальний запас ходу електро-вантажівки. Іншими словами, не існує ділянки маршруту, на якій електро-вантажівка повністю розрядилась та водій був не в змозі доїхати до найближчої заправки.

Висновки

В даній роботі було запропоновано та описано спосіб побудови маршрутів для водіїв електровантажівок.

Список використаних джерел

1. E. Lawler, J. Lenstra, A. RinooyKan, and D. Shmoys, The Traveling Salesman Problem: a guided tour of combinatorial optimization, Vol. 3 (Wiley New York, 1985).
2. D. Applegate, R. Bixby, V. Chvatal, and W. Cook, The traveling salesman problem: a computational study (Princeton University Press, 2011).
3. M. Nielsen and I. Chuang, Quantum Computation and Quantum Information, 2nd ed. (Cambridge University Press, 2010).
4. Левитин А.В. Глава 3. Метод грубой силы: Задача коммивояжёра // Алгоритмы. Введение в разработку и анализ — М.: Вильямс, 2006. — С. 159–160. — 576 с.

УДК 004.822

БЕВЗ Д.О.,
БАКЛАН І.В.

СИСТЕМИ САМОСТІЙНОГО ЗДОБУТТЯ ЗНАНЬ НА ОСНОВІ СЕМАНТИЧНИХ МЕРЕЖ

У даній статі запропоновано та описано спосіб покращення існуючої системи самостійного здобуття знань на основі використання семантичних мереж та їх візуальному представленню. Наведено принцип роботи системи, та обґрунтовано вибір підходу розв'язку поставленої задачі.

This article proposes and describes how to improve the existing system of independent knowledge acquisition based on the use of semantic networks and their visual representation. The principle of the system's operation is stated, and the choice of the approach to the solution of the problem is justified.

1. Вступ

Процес навчання у людей починається з самого народження. Різного роду знання ми отримуємо завдяки, власному чи перейнятому від інших, досвіду. Ці здобутки залишаються в нашій пам'яті в тому чи іншому вигляді. Причому у кожної людини спосіб мислення, ідеї, основні концепції та взаємозв'язки між ними різні і залежать в повній мірі від когнітивної структури особистості. Саме ця структура має принципове навантаження в плані побудови моделі світу вцілому.

2. Постановка проблеми

Для здобуття статусу спеціаліста певної предметної області необхідні відповідні теоретичні та практичні навички. На жаль рівень підготовки спеціалістів у вищих навчальних закладах не відповідає існуючим потребам на ринку. Щоб стати конкурентоспроможним необхідно приділяти багато часу самонавчанню. Можна виділити основні проблеми в здобутті необхідних знань:

- розпливчатість кордонів необхідних знань;
- неточність при оцінені свого поточного рівня знань та прогресу вцілому;
- велика кількість інформації, що підлягає аналізу;
- конфлікт когнітивних структур авторів та споживачів навчального матеріалу.

3. Мета статті

Метою даної статті є демонстрація способу повного або часткового вирішення проблем перечислених раніше та полегшення здобуття нових знань.

4. Основна частина.

Хоча сприйняття та представлення інформації у кожної людини різні, проте сама структура знань у всіх однакова. Для початку пропоную структурувати знання, які представлені у певній предметній області. Як засіб збереження та візуалізації знань будемо використовувати семантичні мережі, а саме їх представлення у вигляді семантичних дерев.

Семантичні мережі по своїй суті - орієнтовані графи з позначками на дугах. Апарат даних мереж є природною формалізацією асоціативних зв'язків, яким користується людина при добуванні якихось нових фактів з наявних. Побудова мережі сприяє осмисленню інформації та знань, оскільки дозволяє встановити суперечливі ситуації, недостатність наявної інформації й т.п. Крім того завдяки легкій реструктуризації процес додавання нової інформації, нових знань не викликає жодних проблем.

Семантичні дерева, як зазначалося раніше, це наочне відображення, відповідних їм, семантичних мереж(рис. 1). Вершини побудованих графів можуть кількісно характеризуватися ступенем, що вказує на кількість сусідніх вершин і вимірює зв'язність вершини. Оскільки вузли являють собою слова(чи словосполучення), то кількість сусідніх вершин також

забезпечує якісну статистику у різноманітності використання слів відповідно до їх розташування в базовій семантичній мережі. Вершини з високими показниками вказують на високу диференційовану залежність і, як правило, вказують на місцеві вузли в семантичній мережі. Вибравши зацікавивший нас напрям навчання, вершина може бути використана для вивчення сусідніх вузлів. Для відстежування прогресу доцільно створювати підграфи, котрі міститимуть кластер вивченої людиною інформації.

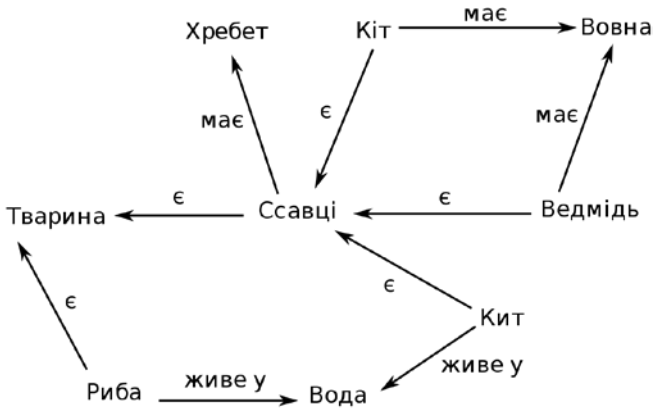


Рис. 1. Спрощений приклад семантичної мережі

Ребра являють собою відносини між двома сусідніми вершинами. Вони можуть бути зваженими відповідно до статистичних величин відповідних вершин. Ребра показують, що дві вершини з'єднані між собою певним словосполученням.

Важливий етап пошуку шляхів у графі. Шлях описує набір пов'язаних ребер, які з'єднують дві вершини одна з одною. Кожен шлях має довжину, що служить його кількісною характеристикою. Шляхи можуть бути використані для визначення усіх можливих (найкоротших) з'єднань між двома вершинами або вузлами, які допомагають з'ясувати, як вони з'єднані одне з одним. Якщо між двома вузлами існує шлях, ми можемо вказати для семантичної мережі, що існує або пряме, або непряме семантичне відношення на словесній основі. Шлях між двома вершинами дасть інформацію, щодо порядку вивчення і можливих шляхів поглиблення знань. На даному етапі вибір метода такого пошуку не є важливим.

Наступним кроком буде переведення усіх знань, здобутих людиною в певних предметних областях, у семантичні мережі та відповідні їм графи, що будуть по суті підграфами повних графів предметних областей. Стає доступною можливість, як програмно так і візуально, порівнювати семантичні мережі повного комплексу знань зі знаннями окремих людей. Після порівняння наявно буде показано, які сутності чи зв'язки між сутностями ще не вивчені, не встановлені. Обравши, ще не здобуті, знання будується шлях котрий буде направляти людину у її прагненні до навчання.

Висновки

В даній роботі було запропоновано та описано спосіб покращення існуючої системи самостійного здобуття знань. Порівнюючи відповідні мережі предметних областей з власними кожна людина, без проблем, може визначити яких знань їй не вистачає і як це виправити. В процесі навчання їх власні семантичні мережі будуть рости, змінюватися показуючи прогрес. Крім того такими мережами зручно ділитися з іншими, наприклад прикріплювати до резюме.

Список використаних джерел

1. Електронний ресурс: http://koi.tspu.ru/koi_books/gorchakov
2. Aggarwal, C.C. Data Mining. / C.C. Aggarwal // Cham: Springer, Int. Publ. – Switzerland. – 2015. – 734p.
3. Електронний ресурс: https://uk.wikipedia.org/wiki/Семантична_мережа
4. Електронний ресурс: https://en.wikipedia.org/wiki/Graph_(discrete_mathematics)
5. Басакер, Р. Конечные графы и сети / Р. Басакер, Т. Саати. - М.: [не указано], 1996. - 123 с.

УДК 004.043

ЗІНЧЕНКО Л.В.,
КОСТИЧЕВА К.Ю.

ПРАКТИЧНЕ ЗАСТОСУВАННЯ МЕТОДІВ КЛАСТЕРИЗАЦІЇ У КОМПЛЕКСІ ЗАДАЧ ІНФОРМАЦІЙНОЇ ПІДТРИМКИ КОМУНІКАЦІЇ МЕНТОРІВ ТА УЧНІВ

У статті представлений підхід, що дозволяє знаходити користувачів зі схожими поглядами. Даний підхід базується на використанні методів кластеризації у багатовимірному просторі. Визначено критерії кластеризації та їх міра. На підставі проведеного експериментального аналізу ефективності методів обрано найбільш підходящий метод кластеризації відповідно до поставленої задачі.

The article presents an approach that allows users to find users with similar views. This approach is based on using clustering methods in a multidimensional space. Clustering criteria and their extent are determined. Based on the experimental analysis of the efficiency of the methods, the most suitable clusterization method was chosen according to the task.

Ключові слова: МЕТОДИ КЛАСТЕРИЗАЦІЇ, FOREL, K-MEANS, ОДНОДУМЦІ, МЕНТОР, УЧНЬ.

1. Вступ

Багато молоді, в тому числі студенти, виконуючи домашні завдання, лабораторні роботи тощо потребують допомоги когось, хто роз'яснить їм незрозумілий матеріал. Для цього існують репетитори або ментори.

Ментор – це людина, що самостійно визначає вектор навчання та його інтенсивність. Задач ментора полягає у тому, щоб допомагати тоді, коли допомога необхідна. Як їх знайти? Якщо раніше пошук здійснювався по газетах чи оголошення на зупинках, зараз він відбувається в Інтернет. Наразі в Україні дуже мало сайтів, які надають можливість саме учням знайти ментора або залишити заявку про допомогу, а також об'єднатись з іншими учнями-однодумцями та навчатись разом, а менторам надавати свої послуги не вивішуючи перед цим оголошення десь на вулиці чи в газеті, а просто зареєструвавшись на сайті. Тобто це не просто пошук репетитора по предметній області, це пошук по задачі чи проблемі, яку учень не може вирішити чи зрозуміти, це можливість знайти ментора, який компетентний саме у конкретному питанні.

Авторами створене WEB-застосування для комунікації менторів та учнів, що має вирішити цю проблему. Воно надає такі основні функціональні можливості, що розділяються за користувацькими ролями. Учні мають можливості: керування заявками (створення заявок про допомогу, позначення заявок як вирішені або скасовані); заповнення анкети для пошуку однодумців та пошук однодумців; оцінювання ментора після отримання від нього допомоги. У свою чергу ментори мають можливість: переглядати заявки від учнів та відгукуватись на них; ведення ненаукових статей (невелика підбірка інформації, що висвітлює певну проблему та пропонує її рішення або навчальний матеріал), що можуть продемонструвати його кваліфікованість у певних питаннях. Функціонал, що є спільним для обох ролей: спілкування у чаті, заповнення особистої інформації у профілі та оцінювання публікацій менторів.

У даній роботі розглядається використання методів кластеризації для пошуку однодумців та вибір алгоритму кластеризації для поставленої задачі.

2. Пошук однодумців

Однодумці – група людей зі схожими поглядами на життя, переконаннями, схожими способами мисленням, стилем життя (якоюсь мірою схожі/однакові) або люди, що займаються однією справою. Однодумцям має бути легше знаходити спільну мову під час навчання, обміну досвідом, обміну навчальними, дидактичними чи розвиваючими матеріалами та розмов на певну предметну

тематику. Тож крім спільних поглядів на життя будуть враховуватись теги-категорії, які кожен користувач вводить у своєму профілі. Кожна категорія має оцінку (вагу), яка буде відповідати позиції тегу у списку тегів користувача. Тобто якщо користувач додав тег “А” першим, то його вага буде 1, а якщо користувач додав тег “Б” другим, то його вага буде 2. Не дивлячись на те, що користувач може вказати багато тегів (до 20), проте ми будемо враховувати лише перші 5, вважаючи, що це найважливіші теми користувача і є його основними, з якими він працює у нашій системі. Теги користувачів будуть враховуватися автоматично при проходженні анкетування, що дуже зручно для користувачів (учнів, які проходитимуть опитування), адже їм не доведеться задумуватися про те, щоб ще раз вводити тег-категорії і щоб серед знайдених однодумців були саме ті ментори та учні, які мають спільні цілі у нашій системі.

Кластерний аналіз можна застосовувати у задачах пошуку команди людей, спеціалістів для реалізації спільних ідей на основі даних, що генеруються у соціальних мережах. Кластеризація використовується у інтелектуальних системах семантичного аналізу публічних повідомлень для попередження незаконних дій групи людей [1], а також у сфері психології для визначення класів даних та їх структури[2].

Для знаходження однодумців було вирішено використати метод інтелектуального аналізу даних, такого як кластеризація. Цей метод використовує дані, отримані від кожного учня у результаті проходження ним анкетування. Анкета складається з певної кількості питань, що асоціюється з певною характеристикою. Кожному питанню ставиться у відповідність числове значення певної характеристики. Важливо зазначити, що в залежності від поставлених питань можна змінювати критерії пошуку однодумців. За результатами анкетування для кожного користувача потрібно знайти, наприклад, 10 однодумців, тобто поділити всіх користувачів на групи – кластери.

Отже, маємо множину користувачів з відомим набором характеристик та відомою мірою кожної характеристики. Ціль - для кожного учня знайти набір (групу) з G інших користувачів (менторів та учнів) з найбільш схожими значеннями цих характеристик, тобто однодумців.

Так як кількість кластерів заздалегідь може бути невідома, то буде використано 2 алгоритми кластеризації: FOREL [3], якщо необхідно визначити кількість кластерів під час вирішення завдання, і K-MEANS [4], якщо кількість кластерів заздалегідь відома.

Використання двох алгоритмів в різних ситуаціях дозволяє поліпшити якість отриманих результатів.

3. Постановка задачі

Задача пошуку однодумців може бути зведена до задачі кластеризації. Сформуємо математичну модель даної задачі.

Введемо наступні позначення:

– кількість користувачів – N;

– кількість запитань у анкеті – m;

– максимальна кількість користувачів у кожному кластері – G;

– кожному запитанню в анкеті поставлено у відповідність кількісна оцінка x_j; залежно від відповіді ця оцінка може приймати значення від a_j до b_j.

З урахуванням цього кожен користувач представляє собою точку у m-вимірному просторі (m-вимірний вектор).

Отже маємо N точок $x^1 \dots x^N$, де $x^k \in R^m$, $k = 1,\dots, N$.

Кожна точка має обмеження вигляду:

$$a_j \le x_j^k \le b_j,\ k = \overline{1, N},$$
$$j = \overline{1, m};$$
$$\forall x_j^k, a_j, b_j \ge 1.$$

Для заданого елемента x^0 потрібно знайти найбільш подібні, що задовольняють умові:

$$\sum_{j=1}^{m} \left(\frac{(x_j^k - x_j^0)^2}{(b_j - a_j)^2} \right) \le k,$$

де k - заданий параметр, що визначає ступінь «схожості».

4. Методи. Порівняння алгоритмів вирішення задачі

Прийнято ділити всі алгоритми кластеризації на ієрархічні і неієрархічні. Розподіл цей відбувається по отриманих на виході даних. Ієрархічні алгоритми на виході

видають певну ієрархію кластерів, і ми можемо вибрати будь-який рівень цієї ієрархії для того, щоб інтерпретувати результати алгоритму[5]. Неієрархічні – це, фактично, всі алгоритми, які на виході ієрархію не видають (або вибір інтерпретації відбувається не за рівнем ієрархії).

Неієрархічні алгоритми за методом виконання поділяють на:

- ітеративні;
- модельні;
- концептуальні;
- мережеві.

Алгоритм K-MEANS належить до ітеративних алгоритмів. Ітеративні алгоритми називаються так тому, що ітеративно перерозподіляють об'єкти між кластерами[6]. Ієрархічні методи відрізняються від ітеративних тим, що вони не вимагають попереднього визначення числа кластерів. Тобто для того, щоб можна було використовувати ітеративний алгоритм необхідно мати гіпотезу про найбільш ймовірну кількість кластерів. Головна думка – мінімізація різниці між елементами кластера і максимізація відстані між кластерами[7].

Загальноприйнятої класифікації методів кластеризації не існує, але можна виділити ряд таких методів:

- графові методи кластеризації;
- ієрархічна кластеризація (таксономія);
- статистичні методи кластеризації.

Алгоритм FOREL відноситься до графових методів, а K-MEANS належить до статистичних методів кластеризації. Основна ідея метода FOREL полягає у пошуку об’єктів, що знаходяться у сфері з певним радіусом, який задається на початку роботи алгоритму[1].

5. Проведення експериментів. Результати.

Розглянемо алгоритми FOREL та K-MEANS при $N = 1000$ точок (користувачів), $m = 20$ питань, відповіді на питання матимуть оцінку від $a_j = 1$ до $b_j = 5$. Будемо змінювати кількість варіантів відповідей, щоб побачити динаміку, та як від цієї кількості залежать час виконання алгоритмів.

Алгоритм FOREL дуже чутливий до радіуса, який задамо на початку, тож ми провели низку експериментів, що дізнати найкращий радіус при різних кількостях варіантів відповідей. Для нас найзручніший радіус той, який дасть приблизно по $G = 10$ точок у кластері. Оскільки ми проводимо експерименти з 1000 точок, то підходяща кількість кластерів визначається за формулою (1) і дорівнює 100 кластерам. На графіку (рис. 1) можемо бачити, при якому радіусі отримаємо 100 кластерів. Визначивши радіус для алгоритму FOREL, приступимо до експериментів з визначення часу двох алгоритмів в залежності від кількості відповідей на питання (якщо відповіді 2 - то максимальна оцінка питання 2, якщо відповіді 3 - то оцінка 3 і т. д.).

Після визначення радіуса відповідно до максимальної оцінки питання та необхідної кількості кластерів, проведено випробування обох методів та визначено середній час їх виконання. Отримані результати показали, що при оцінках 2 та 3 швидкість виконання алгоритмів відрізняється у декілька разів(FOREL меньше), але значно не коливається. При оцінці, рівній 4, можна помітити спад часу виконання обох алгоритмів, а про оцінці 5 відбувається його зростання, при тому, що відношення швидкостей виконання залишається майже незмінним.

Тобто FOREL з добре підібраним радіусом набагато швидше ніж K-MEANS, проте якщо змінити хоч один з наступних параметрів: кількість точок на вхід, розмірність точок (кількість питань в анкеті), максимальна оцінка (залежність від максимально оцінки показана на рис. 1), - то знову доведеться підбирати новий радіус, щоб тримати результати, що нас влаштовують, а саме топ 10 однодумців. На противагу цьому K-MEANS більш гнучкий, і при динамічній зміні користувачів у WEB-застосуванні не потрібно буде щось міняти в коді, а кількість кластерів, що подається на всіх буде розраховуватися за формулою:

$$k = \frac{N}{10}, \qquad (1)$$

де N - кількість точок і $N \geq 10$.

Отже, з отриманих результатів експериментів можна зробити висновок, що найменший час виконання саме при

кількості відповідей на питання 4, тож будемо підбирати питання враховуючи це.

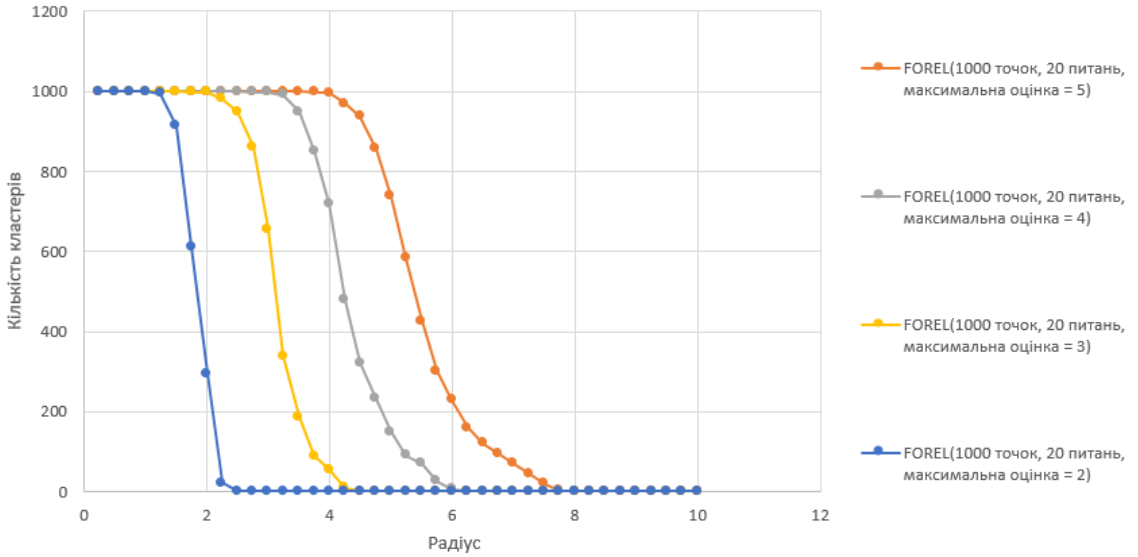


Рис. 1 - Графік залежності кількості кластерів від радіусу для алгоритму FOREL

6. Висновки

Отже, алгоритм FOREL з підібраним радіусом проявив себе краще в порівнянні з K-MEANS, при кількості точок 1000. Його час виконання менший у 5 разів. Проте при зміні вхідних параметрів, в тому числі кількості точок (що в нашому випадку критично), потрібно шляхом експериментів підбирати радіус для алгоритму FOREL, що говорить про те, що він не є гнучким. Зважаючи на це, беремо алгоритм K-MEANS та інтегруємо його у наше WEB-застосування. Під час створення анкети будемо орієнтуватися на те, що найбільш прийнятна кількість варіантів відповідей для кожного питання 4.

Отже, був реалізований спеціальний модуль “Пошук однодумців” у комплексі задач інформаційної підтримки комунікації менторів та учнів, який використовує алгоритм K-MEANS, а також як вхідні дані була використана реальна анкета з 4-ма варіантами відповідей з оцінкою від 1 до 4.

Таким чином, реалізована можливість для учнів знаходити собі менторів-однодумців, а також просто однодумців, що цікавляться такими самими предметними областями, що і учень, та мають схожі погляди, думки, поведінку в певних ситуаціях та спосіб життя, та в подальшому спілкуватися зі знайденими однодумцями. За результатами експериментів краще себе показав K-MEANS, тож він більше нам підійде.

Список літератури

1. Охапкина Е. П., Охапкин В. П.. Подходы к кластеризации групп социальной сети // Компьютерные исследования и моделирование, 2015, Т. 7 № 5 С. 1127−1139.
2. Климчук В.О., Кластерний аналіз: використання у психологічних дослідженнях // Практична психологія та соціальна робота. – 2006. – №4. – С. 30−36.
3. Гитис Л. Х., Статистическая классификация и кластерный анализ, М. : Горная книга, 2003.
4. Мандель И.Д., Кластерный анализ – М.: Финансы и статистика, 1988. – 176 с.
5. A.K. Jain and R.C. Dubes. Algorithms for Clustering Data. Prentice Hall, 1988.
6. Adam Coates and Andrew Y. Ng. Learning Feature Representations with K-means // Stanford University, 2012,
7. Якимець Р. В., Методи кластеризації та їх класифікація // Міжнародний науковий журнал. – 2016. Т. 2, № 6.

УДК 004.94

МІРОШНИК О.С.

МЕТОД КЛАСИФІКАЦІЇ ТА ТЕГУВАННЯ РЕКЛАМИ З УРАХУВАННЯМ КОНТЕНТУ, ЩО ПЕРЕГЛЯДАЄ КОРИСТУВАЧ

Розглянуто практичне застосування методів Natural-language processing на прикладі задачі визначення рекламних рекомендацій користувача системи Octogen. Демонструється використання баєсівського класифікатору, описано алгоритм обробки тексту та метод визначення категорій вподобань для отримання найбільш релевантної та змістовної реклами.

The practical application of Natural-language processing methods is considered on the example of the task of determining the advertising recommendations of the user of the system Octogen. Demonstrates the use of the Bayesian classifier, describes the algorithm of text processing and the method of determining the categories of preferences for the most relevant and meaningful advertising.

1. Вступ

Задача формування рекламних рекомендацій користувача ґрунтується на його вподобаннях. Для отримання такого роду інформації існує декілька методів: вимагати від користувача введення даних самому або аналізувати історію його дій та зміст контенту, що він найчастіше переглядає, шукає тощо. У нашій системі у якості даних, для аналізу використовується зміст веб-сторінок, які відвідував користувач під час перебування у мережі Інтернет.

Одним з рішень поставленої задачі класифікації та співставлення даних користувача з певними категоріями та тегами реклами є використання нейронних мереж та NLP (Natural-language processing). Необхідно створити модель, яка на вхід отримує зміст веб-сторінок, вибирає ключові слова та визначає категорію, що підходить найбільше і таким чином навчатися.

2. Вхідні дані

Навчання моделі розбито на два етапи: навчання на основі підготованих наборів даних та навчання на основі інформації, яку ми отримуємо безпосередньо від користувача.

На першому етапі використовувалися набори даних з веб-ресурсу “20 Newsgroup”[1] на основі новин різних категорій та власноруч зроблена вибірка статей за категоріями і темами з веб-порталу “Medium”. Таким чином, текст, що надходив на вхід до моделі вже був класифікований і розбитий на категорії. Моделі залишається навчитися розуміти, яким категоріям відповідають наявні у тексті слова та їх комбінації.

Другий етап передбачає отримання даних безпосередньо від користувача. Для цього використовується веб-розширення для браузеру, яке дозволяє отримувати доступ до змісту веб-сторінок, що переглядає користувач, і відправляти на сервер необхідні дані. Більшість веб-сторінок мають meta-теги з переліком ключових слів. Ці слова можна використовувати для визначення категорії контенту. Інформація, яку переглядає користувач, знаходиться у тезі body. На основі довжини тексту, параметрів інших тегів можна визначити головний зміст сторінки та використовувати цю інформацію у якості вхідних даних до нашої моделі.

3. Обробка вхідних даних

Розуміти людську мову для комп’ютера важка задача, оскільки потрібно розрізняти різні сутності та смислові конструкції. Аби нейронна мережа могла розуміти слова та використовувати алгоритми машинного навчання необхідно перетворити текст до більш логічного виду - до числового представлення, а саме до числового вектору. Для цього використовується модель bag-of-words, яка рахує кількість зустрічей кожного зі слів у тексті. Проблемою на цьому етапі є те, що перевага віддається великим документам, тож аби уникнути цього ми використовуємо статичний показник для оцінки важливості слів TF-IDF (TF – term frequency, IDF – inverse document frequency) аби нормалізувати вибірку.

Більшу вагу TF-IDF отримують слова з високою частотою появи в межах документа та низькою частотою вживання в інших документах колекції. Відношення числа входжень обраного слова до загальної кількості слів документа:

$$TF = \frac{n_i}{\sum_i n_i}$$

де n_i є число входжень в документ, а в знаменнику – загальна кількість слів в документі.

Інверсія частоти, з якою слово зустрічається в документах колекції визначається наступним чином:

$$IDF = \log \frac{|D|}{|(d_i \supset t_i)|}$$

де $|D|$ – кількість документів колекції, $|(d_i \supset t_i)|$ – кількість документів, в яких зустрічається слово t_i (коли $n_i \neq 0$).

$$TFIDF = TF \cdot IDF$$

4. Баєсівська класифікація

Існують різні алгоритми класифікації тексту[2]. У своїй реалізації ми використовуємо наївну баєсівську класифікацію – ймовірнісну класифікацію, як передбачає розподіл за списком визначених класів. Це дозволяє нам розраховувати імовірність присвоєння певної категорії до тексту. Формула для вирішення задачі класифікації задається наступним чином:

$$P(C|D) = \frac{P(C|D)P(C)}{P(D)}$$

де C – це клас, а D – це сам документ.

За допомогою цієї теореми визначається імовірність присвоєння класу C до документа D за умови, що ми знаємо імовірність відношення D до C і незалежні імовірності для C і D.

Під словом “наївна” розуміється припущення, що всі слова у документі не залежать одне від одного. Таким чином ми можемо визначити імовірність як суму імовірностей для кожного окремого слова w_i:

$$P(D|C) = P(w_1, \dots, w_i|C) =$$
$$= P(w_1|C) \dots (w_i|C)$$

Існують різноманітні баєсівські класифікатори. Вони роблять припущення стосовно розподілення імовірностей для різних класів. Мультиноміальна класифікація враховує ще й частоту входження. Імовірність присвоєння класу для слова можна виразити наступним чином:

$$P(w_i|C) = \frac{count(w_i, C) + \propto}{count(C) + \propto \cdot V}$$

де $count(w_i, C)$ – кількість того, скільки слово w_i було віднесено до класу C на тренувальних даних, $count(C)$ – кількість всіх слів, що віднесені до класу C, V – кількість унікальних слів у тренувальних даних, $\propto$ - параметр, що запобігає імовірності 0.

5. Висновки

Класифікація та аналіз великої кількості текстів для визначення рекламних рекомендацій є актуальною і важкою задачею, яка наразі може бути вирішена за допомогою сучасних методів NLP та імовірнісних класифікаторів. Користувач безпосередньо приймає участь у навчанні моделі за рахунок інформації отриманої з переглянутих веб-сторінок. Отримана класифікація дозволяє визначати вподобання користувача та, ґрунтуючись на цьому, отримувати релевантні рекламні пропозиції.

Список використаних джерел

1. Rennie J. Набори даних з порталів новин [Електронний ресурс] / Jason Rennie. – 2008. – Режим доступу до ресурсу: http://qwone.com/~jason/20Newsgroups/.
2. Гавриленко О. В. Огляд та аналіз алгоритмів TEXT MINING / О.В.Гавриленко, Ю.О.Олійник, Г.В.Ханько. // Управління проектами, системний аналіз і логістика. – К.: НТУ, 2017. – Вип.
3. T. S. Guzella and W. M. Caminhas, “A review of machine learning approaches to spam filtering,” Elsever, Expert System with Applications,2009.– с. 11–20.
4. A. Markov and M. Last, “A simple, structure-sensitive approach for web document classification”, in Atlantic Web Intelligence Conference – AWIC, 2005. – с. 293–298.
5. Vinciarelli A., “Noisy Text Categorization, Pattern Recognition”, 17th International Conference on (ICPR), 2004. – с. 554-557.
6. C.-H. Lee and H.-C. Yang, “Construction of supervised and unsupervised learning systems for multilingual text categorization,” Expert Systems with Applications, 2009.– с. 2400–2410.

УДК 004.023

КАТЮЩЕНКО Д.О.,
ОЛІЙНИК Ю.О.

ДОСЛІДЖЕННЯ МЕТОДІВ ПОПЕРЕДНЬОЇ ОБРОБКИ ТА ВЕКТОРИЗАЦІЇ ТЕКСТОВИХ ДОКУМЕНТІВ

Дана стаття присвячена аналізу проблем представлення текстової інформації в векторному вигляді для застосування в методах text mining. Приведені існуючі підходи до зняття морфологічної омонімії. Запропонований алгоритм автоматичної класифікації текстової інформації на базі таких методів та підходів як, алгоритм Вітербі для зняття омонімії, критерій спільної інформації для визначення зв'язаних слів речення, TF-SLF в якості методу визначення ваг ознак документу та класифікатор Роші.

This article is devoted to the analysis of the issues regarding representation of text information as a vector form for using them in text mining methods. The existing approaches to the removal of morphological homonymy are presented. There is proposed the algorithm for automatic classification of textual information based on such methods and approaches as the Viterbi algorithm for the removal of homonymy, the common information criterion for the definition of related word in the sentences, TF-SLF as a method for determining the weight of the document terms and the Roshi classifier in the article.

1. Вступ

Враховуючи постійне зростання об'ємів інформації, розвиток методів автоматичної обробка текстової інформації є надзвичайно актуальним. Одною із задач інтелектуального аналізу текстів є їх класифікація на задані категорії, яка знаходить використання в різних сферах людської діяльності. Першим етапом автоматичної класифікації є представлення документу в зручному для обчислень вигляді. Зазвичай - у вигляді вектору ознак (термів). Підготовка та векторизація є технічно та обчислювально трудомісткими етапами інформаційного аналізу. Їх задача формальне представлення текстів зі збереженням їх змісту. В даній статі буде розглянуто основні способи формалізації текстових документів, проблеми які при цьому можуть виникнути та запропоновано алгоритм автоматичної класифікації.

2. Способи формалізації текстових документів

Основні відомі способи формалізації текстових документів для подальшої автоматичної обробки є:

1. Вектор ознак (VSM - VectorSpacemodel).[1] Текстовий документ представляється у вигляді вектору, кожна координата якого відповідає вазі терма (зазичай частоті зустрічання цього терма в текстовому документі). По завершенню обробки вибірки отримаємо прямокутну матрицю розмірності $m \times n$, де n - кількість документів, m - кількість слів в колекції. Таке представлення тексту не потребує використання додаткових попередніх аналізів;

2. Поліграмна модель.[1] В поліграмній моделі текст представляється через n-грами - послідовності n символів, які йдуть один за одним. Вважається, що частота з'явлення n-грами в тексті несе важливу інформацію про властивості документа. Крім того максимальна кількість n-грам постійної довжини для конкретної мови є фіксованою та не залежить від об'єму навчальної вибірки. Перевагами поліграмної моделі є те, що немає необхідності додаткової лінгвістичної обробки тексту, фіксована розмірність векторів та простота отримання векторного опису тексту. Однак таке відображення змісту тексту не завжди є адекватним (такий підхід погано відображає зміст невеликих текстів), тому модель більше підходить для визначення мови ніж для класифікації по тематиці;

3. Лінгвістична розмітка.[1] Представляє текстові документи з

синтаксичними характеристиками, тобто задає інформацію про лінгвістичні одиниці безпосередньо в тексті в формі розмітки на таких мовах, як SGML або XML. Для приведення тексту до такого представлення використовують синтаксичний або морфологічний аналіз;

4. Дерева залежностей.[1] Граматика залежностей передбачає, що речення тексту можна структурувати у вигляді дерев залежностей, в яких слова зв'язані орієнтованими дугами, які визначають синтаксичний зв'язок між головним та підлеглим словами.

3. Проблеми, які виникають при попередній обробці текстів

Морфологічна омонімія [2]- збігання однієї або декількох граматичних форм слів в написанні або мовленні, які належать різним частинам мови. Наприклад, *рукав* (елемент одягу) — *рукав* (річки).

Синтаксична омонімія [2] - неоднозначність, яка виникає через неясність синтаксичних зв'язків між словами у реченні.

Вирішення кореферентності [2] - визначення відносин між компонентами речення, в якому слова посилаються на одні й ті самі об'єкти.

4. Способи зняття морфологічної омонімії

Існує 3 підходи для зняття морфологічної омонімії:

1. детермінований підхід - заснований на локальному та глобальному синтаксичному розборі, синтаксичних словниках та на правилах узгодження слів. Такий підхід потребує великих затрат часу та ресурсів.

2. статистичний підхід - використовує статистику спільної зустрічаємості слів в великих корпусах, омонімія в яких знята заздалегідь. Виконується на етапі морфологічного аналізу.

3. Гібридний підхід - поєдную в собі обидва підходи. Зазвичай він заснований на статистичних методах та оснащений невеликою кількістю правил.

Найпоширенішими статистичними методами зняття морфологічної омонімії є HMM (прихована Марківська модель), MEMM (Марковські моделі із максимальною ентропією) та алгоритм Вітербі[3].

Для розробки алгоритму класифікації був обраний алгоритм Вітербі, який полягає в наступному:

Необхідно знайти c з максимальною повною ймовірністю:

$$argmax_{c \in C} \prod_{t \in N} (c_t | c_{t-1}) * T(w_t, c_t), \quad (1)$$

де $T(w, c) = \frac{c_t(w,t)}{c(w)}$;

$c_t(w, t)$– кількість слів w, які мають тег t;
N - кількість слів в реченні;
c – послідовність тегів;
w - слово, яке розглядається;
t - тег відповідного слова.

4. Алгоритм автоматичної класифікації

Нижче приведений метод алгоритм автоматичної класифікації текстів з використанням гібридного методу зняття морфологічної омонімії (на основі алгоритму Вітербі та правилах визначення лексико-граматичних класів на основі квазізакінчень). В якості методу визначення ваг ознак документу обрано TF-SLF [3], а класифікатор - Роші [4].

Крок 1: Формування D - масиву документів навчальної вибірки та C - масиву категорій (класів).

Крок 2: Токенізація– розбиття$d_i \in D,\ i = \overline{1, |D|}$на речення - $s_{ij} \in D_i, j = \overline{1, |D_i|}$.

Крок 3: Видалення зайвих символів, семантично нейтральних слів, таких як сполучники, прийменники, артиклі тощо. (на базі заданого словника для зменшення обчислень).

Крок 4: Морфологічний аналіз та тегування.

Визначення лексико-граматичного класу слова$w \in s_{ij}$на базі словника квазізакінчень. Спосіб подання граматичної інформації не за повним словом, а за кінцевою послідовністю літер дозволяє значно скоротити обсяг аналітичного словника. Словникова стаття:

$$< \{F\}_n^1 K_l * \{K_g\} > \quad (2)$$

де $\{F\}_n^1$- ланцюг літер вхідного слова розташований в інверсному порядку;

n- остання літера в слові

K_l- код лексико-граматичного класу вхідної словоформи;

K_g- множина кодів значень граматичних категорій, які визначаються дляK_l.

4.1. в аналізованого слова за словником виділяється квазізакінчення;

4.2. визначене квазізакінчення відсікається;

4.3. визначається наявність у слова інших значень;

4.4. застосування алгоритму Вітербі для визначення частини мови;

4.5. визначається граматична форма, яку потрібно генерувати;

4.6. підраховується номер відповідної граматичної форми;

4.7. вибирається необхідне квазізакінчення зі словника;

4.8. до слова приписується встановлене квазізакінчення.

Крок 5: Видалення з частотного словника тегів з відсотком зустрічаємості менше порогу λ.

Крок 6: Обчислення залежності слів в тексті за критерієм спільної інформації:

$$I(w,v) = log_2 \frac{P(w,v)}{P(w)P(v)}, \quad (3)$$

де $P(w,v)$- ймовірність появи слів w та v разом;

$P(w)$- ймовірність появи слова w ;

$P(v)$ - ймовірність появи v.

Крок 7: Зважування термів:

$$TFSLF_t = TF_t * SLF_t, \quad (4)$$

$$SLF_t = log \frac{|C|}{R_t}, \quad (5)$$

$$R_t = \sum_{c \in C} NDF_{tc}, \quad (6)$$

C - множина категорій в корпусі документів.

$$NDF_{tc} = \frac{df_{tc}}{N_c}, \quad (7)$$

де df_{tc}- кількість документів категорії c в яких хоча б раз зустрічається терм t;

N_c- кількість документів в категорії c.

Крок 8: Формування множини ознак, з урахуванням порогової ваги ознак β.

Крок 9: Розрахунок центроїду категорії:

$$\overline{\mu(c)} = \frac{1}{|D_c|} \sum_{d \in D_c} \overline{v(d)}, \quad (8)$$

де D_c - набір документів з навчальної вибірки, які належать класу c.

Крок 10: Морфологічний та синтаксичний аналіз тексту, який класифікується (d^t) (кроки 2-5).

Крок 11: Зважування термів та проектування вектору документу $\overline{v(d^t)} = \{t_1, \dots, t_{|d^t|}\}$на множину ознак.

Крок 12: Визначення мінімальної відстані між $\overline{v(d^t)}$та класами:

$$Assign \ \ d^t \ \ to \ \ c = \arg\max_c \cos(\bar{\mu}(c^{'}), \bar{v}(d^t)) \quad (9)$$

5. Висновок

Підготовка та векторизація текстових документів є технічно та обчислювально трудомістким етапом інформаційного аналізу. Основні задачі пов'язані з попередньою обробкою текстів це: морфологічна омонімія, синтаксична омонімія та вирішення кореферентності. Їх вирішення значно покращує якість аналізу текстової інформації.

Список літератури

1. Сайт StudFiles. Файловий архів студентів. Лекція "Моделисинтагматическогоподхода" [Електронний ресурс] // Режим доступу: https://studfiles.net/preview/4533558/page:12/

2. БатураТ. В., ЧарінцеваМ. В. Основыобработкитекстовойинформации- Новосибирск: Институт систем информатикиим. А.П. Ершова СО РАН, 2016. - 45 с.

3. Попков М.И. Автоматическая система классификациитекстов для базызнаний предприятия: магистерскаядисертация - М: 2014. - 57 с.

4. Manning D. ChristopherIntroductiontoinformationretrieval: / Christopher D. Manning, PrabhakarRaghavan // CambridgeUniversityPress - 2008.

УДК 004.94

НОСОВ К.С.

МОДЕЛЬ ДАННЫХ КОНТЕНТНОГО РЕСУРСА В ГРАФОВОЙ БАЗЕ ДАННЫХ

В данной статье рассмотрена модель представления контентного сайта в графовой базе данных. Рассмотрена структура данных и связи между ними.

In this article we dig into data model suiteble for storing content site data in graph database. We investigate data model structure and links between data types.

1. Вступление

Контентный сайт - сайт с большим количеством слабо структурированного контента. К примеру сайт новостного портала. В некоторых случаях целесообразно использовать не классический способ представления данных в базе. Мы рассмотрим, как можно организовать данные новостного портала в графовой базе данных Neo4J.

2. Требования к системе публикации

Если CMS изначально не разрабатывалась как устойчивая к нагрузкам и не способная оперировать большим объемом данных, то на определённой стадии жизненного цикла она становиться слишком медленной для конечного пользователя.

Выделим требования к системе публикации контента.

- Гибкость - возможность изменять схемы данных, без изменения программного кода и остановки или перезапуска системы
- Масштабируемость - возможность строить горизонтальные кластеры серверов для увеличения суммарной допустимой нагрузки
- Отказоустойчивость - система должна обеспечивать бесперебойную отдачу контента за приемлемое время без нарушения работоспособности при пиковых нагрузках и адекватное восстановление функционирования в ситуациях превышения допустимых пиковых нагрузок
- Удобство эксплуатации - работа редакторов и людей, обеспечивающих поддержку технической инфраструктуры должна быть автоматизирована, максимально исключить рутинные операции и человеческий фактор

3. Анализ существующих решений.

Рассмотрим CMS Wordpress. На момент написания статьи 30% всех веб-сайтов интернета были написаны на Wordpress. Эта CMS является очень старым, но постоянно развивающимся и обновляющимся программным продуктом. Одним из преимуществ является наличие большой инфраструктуры готовых подключаемых модулей, реализующих самые разнообразные задачи, начиная от добавления блоков текста, заканчивая реализацией многоязычного сайта или превращения сайта в интернет-магазин и интеграцией с бухгалтерским ПО. Это достигается наличием у Wordpress программных интерфейсов и способов создания подключаемых модулей, называемых плагинами, а также нескольких таблиц в БД, называемых terms, taxonomy, term_relationships и term_meta. Схема данных приведена на рис.1. Это единственный способ сохранить и связать данные в Wordpress расширив список доступных для сохранения типов данных.

Wordpress подвержена проблеме масштабирования - по мере того как сайт наполняется контентом, размер БД увеличивается, возрастает количество связей между единицами контента. Все связи хранятся в таблице Posts или в разрозненном виде в четырех вышеозначенных таблицах. В моменты, когда нужно использовать terms или taxonomy, CMS может как сделать новый запрос, так и взять готовое значение из кэша. Что может привести к существенному замедлению работы сайта. Также в большинстве случаев, авторы сторонних плагинов хранят самую разнообразную информацию именно в этих таблицах.

В поле meta_value хранится информация самого разного вида - сериализованный массив с мета-описанием, html, id поля в текстовом виде. Делать одну мультифункциональную таблицу для хранения разных по формату данных является архитектурно неверным решением и нарушением 1-й Нормальной Формы в нескольких случаях к ряду. В случаях, когда нужно организовать

полнотекстовый поиск по данному полю, разработчик ставится в тупик, так как: во-первых, невозможно предвидеть результаты такого поиска - часть данных удовлетворяющих запрос поиска может храниться внутри сериализованного объекта или массива, во-вторых так как часть данных этой таблицы по своей сути хранит идентификаторы в виде строк, скорость выполнения запросов падает катастрофически.

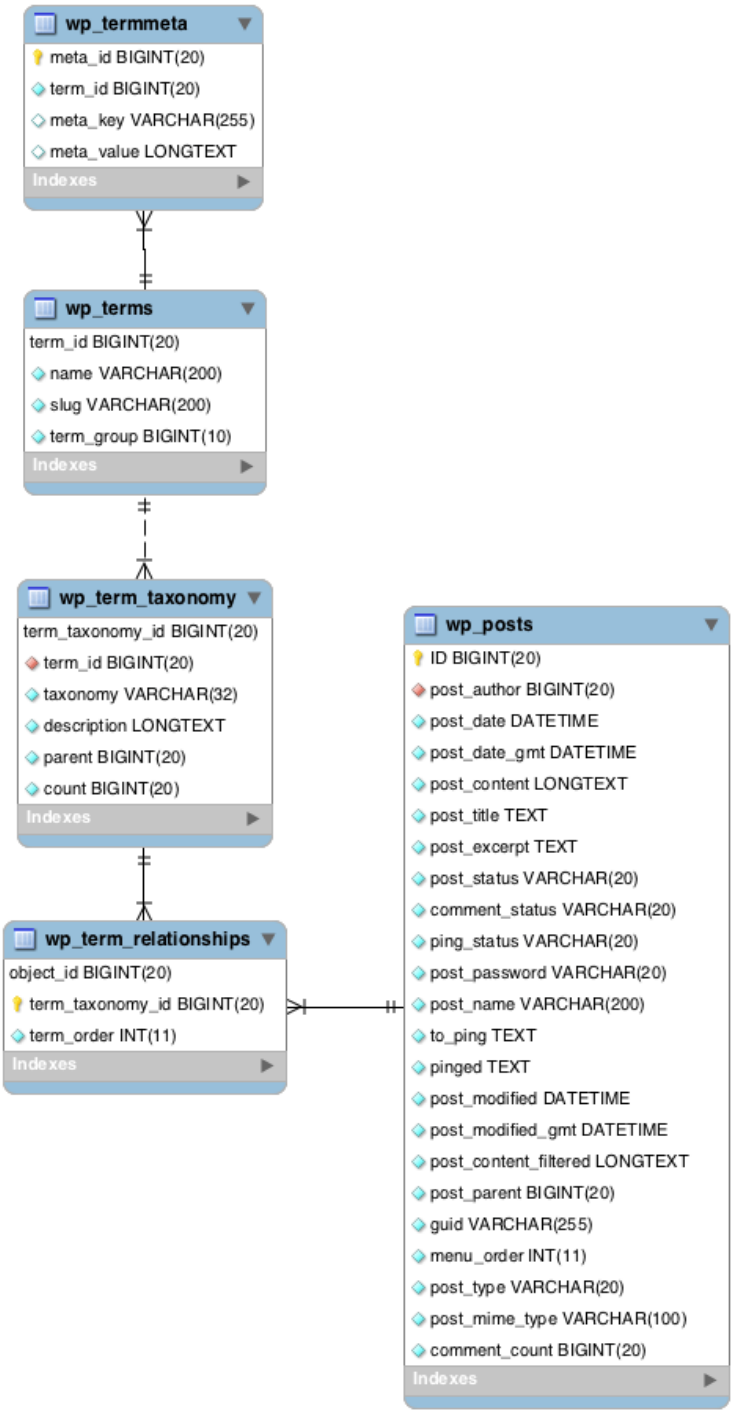


Рис. 1. ER-диаграмма части таблиц базы данных Wordpress

4. Способ представления данных CMS в графовой базе данных

Мы рассмотрим пример организации данных в графовой БД Neo4j на примере модели данных для информационного портала.

В отличие от реляционной БД связи в графовой базе могут иметь типы, атрибуты и значения. Таким образом есть две основные сущности - узел и связь. (рис. 2)

Разберем пример новостного портала. Типы данных которые будут в нем

использоваться: Статья, Автор, Новость, Обзор.

Тип данных - это схема, описывающая повторяемую страницу. Например, все статьи имеют поля “аннотация”, “заголовок”, “дата публикации”. То есть тип данных, это мета описание всех объектов данного типа, в котором хранится список полей и информация о типах полей (типы полей могут быть комбинированными или структурными, например, список допустимых значений).

Тип данных связан с экземплярами данных. Экземпляр — это конечная статья. У нее есть значения, которые комплементарны описанию типа.

Кроме того, в описании типа может быть указана связь с другим типом. Для Статьи это может быть связь с Автором. У каждого экземпляра статьи будет двунаправленная связь с автором. Это значит, что у каждого экземпляра автора будут связи со всеми его статьями.

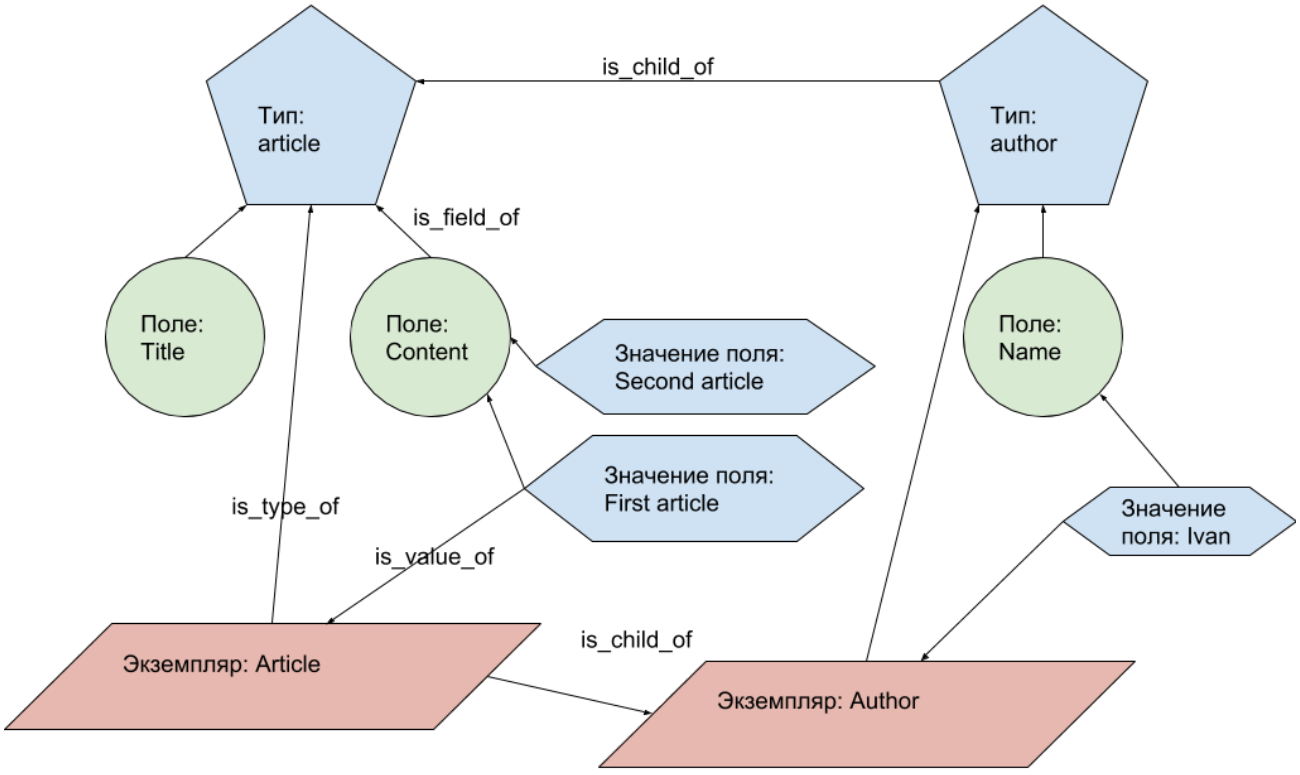


Рис. 2. Структура данных в графовой базе данных

5. Висновки

Использование графов позволяет использовать больше индивидуальных типов данных. Как следствие мы получаем более структурированный контент на сайте. Помимо этого, функции поиска по связям так же упрощают поиск связных данных в базе данных.

Полученная модель хранения данных в базе Neo4J позволяет решать задачу хранения контента для CMS.

Список використаних джерел

1. Ian Robinson, Jim Webber, Emil Eifrem. Graph Databases, 2nd Edition New Opportunities for Connected Data– O'Reilly Media, 2015. – 238 с.
2. Neo4J [Электронный ресурс]. – Режим доступа: https://neo4j.com/docs//
3. Wordpress Codex [Электронный ресурс] – Режим доступа: https://codex.wordpress.org/

УДК 004.056

ЧЕКАНІН О.Ю.
ЖДАНОВА О.Г.

МОДЕЛЬ ЗАГРОЗ ДЛЯ ОЦІНКИ БЕЗПЕКИ АВТОМОБІЛЯ

Робота присвячена визначенню та аналізу атак, виконаних проти реальних автомобілів, що існують на ринку та побудові для них моделі загроз. Використовується методологія моделювання загроз, яка розроблена для аналізу безпеки автомобілям. Результатом є кількісна оцінка загроз, що дозволяє виявити ті, що є найбільш критичними, тобто мають найбільші наслідки з точки зору безпеки компоненти автомобіля.

This article is devoted to the identifying and analysis of attacks against real vehicles available on the market and the creation of a threat model based upon these attacks. The threat modeling methodology is used, which is designed to analyze the safety to vehicles. The result is a quantitative assessment of risks, which allows one to identify those with the most significant consequences and the most critical components regarding the safety of the car.

Ключові слова: безпека автомобіля, електронні компоненти управління, протокол безпеки CAN, модель безпеки.

1. Вступ

Сучасні автомобілі мають багато різних комп'ютерних компонентів, які називаються електронні компоненти управління (ЕКУ) [3]. Все це є результатом того, як автомобілі ставали все більше комп'ютеризованими. Водночас це створило нові ризики для автомобілів як інформаційної системи, пасажирів як користувачів автомобіля та виробників транспортних засобів. Тепер недостатньо забезпечити нормальну роботу усіх компонентів автомобіля та захищати водія разом з пасажирами від ДТП. Сучасні автомобілі привертають увагу зловмисників або хакерів, для яких автомобіль принципово не відрізняється від стаціонарного комп'ютера, банкомата чи смартфона.

Кожен автомобіль містить від 20 до 100 ЕКУ, причому кожен компонент несе відповідальність за одну або декілька функцій транспортного засобу. Найбільш поширеним протоколом обміну інформацією в сучасних автомобілях є протокол CAN - Controller Area Network, який можна вважати стандартом де-факто для обміну інформацією. Цей протокол було створено в середині 1980-х компанією Bosch. Зазвичай в автомобілі присутні дві або три окремі CAN мережі, що працюють з різною швидкістю передачі даних. ЕКУ з'єднані послідовною шиною і кожен компонент "бачить" кожне повідомлення в мережі.

2. Проблеми безпеки протоколу CAN

Протокол CAN має ряд властивих слабких місць, які є загальними для будь-якої реалізації. Ключовими серед них є наступні.

Широкомовна передача. Оскільки пакети CAN передаються фізично та логічно для всіх вузлів, зловмисний компонент у мережі може легко переглянути всі повідомлення або відправляти пакети до будь-якого іншого вузла мережі.

Вразливість до атаки відмова в обслуговувані. Протокол CAN надзвичайно уразливий для атак на відмову в обслуговуванні. На додаток до простих атак у вигляді постійних надсилань пакетів, схема арбітражу CAN на основі пріоритетів дозволяє вузлу встановлювати "домінуючий" стан на шині необмежено і викликати відмову всіх інших вузлів шини CAN.

Відсутність відміток про аутентифікацію. У пакеті протоколу CAN не містяться поля аутентифікації - або навіть будь-які поля ідентифікатора джерела пакета - це означає, що будь-який компонент може надіслати пакет від будь-якого іншого компонента. Це означає, що

будь-який окремий скомпрометований компонент може використовуватися для управління всіма іншими компонентами цієї шини, за умови, що ці компоненти самі не здійснюють захист.

Слабкий контроль доступу. Стандарти безпеки визначають послідовність запитів-відповідей для захисту ЕКУ від певних дій без авторизації.

3. Методика виконання атаки на автомобіль

Критичні з точки зору безпеки атаки проти автомобіля складаються з трьох етапів.

Перший етап полягає в тому, що зловмисник отримує доступ до внутрішньої автомобільної мережі. Це дозволить зловмиснику вводити повідомлення в мережі автомобілів, прямо чи опосередковано контролюючи бажаний ЕКУ. Дослідники з Університету Вашингтона та університету Каліфорнії Сан-Дієго [4] змогли отримати віддалене виконання коду в модулі телематики автомобіля, використовуючи вразливість в програмному забезпечені Bluetooth та скомпрометувати стільниковий модем [3].

Кібер-фізичні напади (атаки, що приводять до фізичного контролю різних аспектів автомобіля), з іншого боку, вимагатимуть взаємодії з іншими ЕКУ. Кібер-фізична атака, як правило, вимагає другого етапу, який передбачає ін'єкцію повідомлень на внутрішню автомобільну мережу, в спробі спілкуватися з критично важливими ЕКУ, такими що відповідальні за керування, гальмування та прискорення.

Третій етап полягає в тому, щоб атакований ЕКУ здійснив певну поведінку, що погіршує безпеку автомобіля. Це передбачає реверс інжиніринг повідомлень у мережі та виявлення точного формату для виконання певних дій.

Існує певний функціонал для запровадження захищеності водія та пасажирів під час водіння, але може бути використано атакуючим для загрози безпеці автомобіля або життю людей в салоні. Наприклад, система виявлення зіткнень для автомобілів Toyota здатна загальмувати автомобіль у разі отримання відповідних пакетів, що свідчать про можливе зіткнення з іншою автівкою. Якщо зловмисник отримає можливість надсилати такі пакети, то результатом буде можливість зупинити автомобіль у будь-який момент.

4. Приклади атак для Ford Escape 2010 та Toyota Prius 2010

Розглянемо деякі атаки, що, проводилися за допомогою пакетів, які надсилаються під час звичайної роботи автомобіля, тобто можуть бути надіслані в будь-який момент без спеціальних передумов.

Показ довільних значень на спідометрі – Форд. Атака дозволяє показати будь-які значення швидкості та обертів двигуна на приборній панелі.

Обмежена можливість керування – Форд. Атака являє собою відмову в обслуговувані. В випадку з Фордом, ЕКУ керування рульовим управлінням вимикається. В такому випадку автомобіль не допомагає водію під час керування, що робить складним поворот колес, а саме неможливо повернути колесо більш, ніж на 45% порівняно з нормальною роботою.

Спідометр – Тойота. Відсутність аутентифікації призводить до того, що неможливо перевірити чи дійсно ЕКУ надіслав пакет. Довільне значення швидкості буде відображатися, надсилаючи постійно пакети з обраним значенням. Пакет необхідно надсилати безперервно, оскільки оригінальний пакет постійно надсилається.

Керування автомобілем – Тойота. Тойота має систему допомоги паркуванню, яка допомагає водію. Досягнувши певних умов зловмисник може керувати поворотом колес при довільній швидкості. Ця атака передбачає безперервне посилання недійсних значень, щоб інші компоненти автомобіля не мали реальних значень.

4. Атаки за допомогою діагностичних пакетів

Для виконання діагностичних операцій, спершу потрібно аутентифікуватися. Аутентифікація полягає в обміну повідомленнями згідно з обраним криптографічним протоколом. Наступним кроком є отримання криптографічного

ключа для розблокування діагностичних операцій.

Усі наступні атаки передбачають створення діагностичної сесії. За дизайном вона необхідна майстрам з авто центру для виявлення проблем в автомобілі та дозволяє робити операції, що недоступні звичайному користувачу.

Застосування гальм – Форд. Діагностичне програмне забезпечення для Форду дозволяю імітувати повністю натиснуті гальма. Також є можливість визначити наскільки сильно треба загальмувати.

Ця атака працює лише, коли автомобіль зупинений. Але після його використання рух стає неможливим, незалежно від того як водій тисне на педаль газу.

Блокування гальм – Форд. Ця атака здійснюється так само, як і попередня, але в даному випадку стає неможливим використати гальма. Ця атака можлива лише при швидкості менш, ніж 5 миль/г.

Вимкнення фар та освітлення – Форд. Діагностичний пакет з ідентифікатором 7E 80 призводить до того, що розподільна коробка вимикається. Разом з цим перестають працювати усі пристрої, що залежать від її роботи. Це фари, внутрішнє освітлення, радіо і т.д. Результатом може бути те, що аварійні фари не будуть спрацьовувати, так само фари не працюватимуть при гальмуванні.

Вимикання двигуна – Форд. Існує діагностичний пакет, що дозволяє вимкнути бажані або всі циліндри двигуна. Це досягається за рахунок того, що нестача палива призводить до ситуації, коли двигун не може працювати. Важливо зазначити, що ця атака не вимагає обов’язкового встановлення діагностичної сесії!

Вимикання двигуна – Тойота. Також існує пакет під час діагностичної сесії, що призводить до припинення постачання палива до усіх циліндрів двигуна. Варто зазначити, що автомобіль має бути припаркованим для здійснення цієї атаки.

Вмикання / вимикання гудка – Тойота. Існує два діагностичних теста, що вмикають та вимикають гудок. Гудок може працювати стільки, скільки буде передаватися спеціальний пакет.

Відкривання / замикання дверей – Тойота. Існують діагностичні пакети для відкривання чи замикання дверей. Це дозволяє зловмиснику віддалено відчинити двері або багажник.

Індикатор палива – Тойота. Існують пакети, що передають кількість палива, що залишилась. Можлива атака, коли водій бачить, що палива вдосталь, хоча воно закінчується. Це може призвести до раптової зупинки під час руху.

5. Модель загроз на базі проаналізованих атак

Моделювання загроз - це процедура оптимізації безпеки мережі шляхом визначення цілей зловмисника та вразливостей, а потім визначення контрзаходів для запобігання чи пом'якшення наслідків загроз для системи.

У цьому контексті загроза являє собою потенційну або фактичну побічну подію, яка може бути шкідливою (наприклад, атакою на відмову в обслуговуванні) або випадковою (наприклад, збою пристрою зберігання даних), що може спричинити небезпеку для захищаємої системи.

6. Стандарти безпеки автомобіля

Разом з все більшим використанням ЕКУ та більшої інтеграції інформаційних технологій в транспортний засіб, збої більше не стосуються зносу або порушенням електричних мереж, а помилкам програмування. Той факт, що електронні компоненти можуть впливати на фізичний світ, змусили виробників транспортних засобів та урядові організації визначити стандарт, згідно з яким виробники автомобілів повинні здійснювати роботу з ризиками. З цією метою в 2011 році стандарт ISO 26262 був прийнятий на базі вже існуючого стандарту IEC 61508.

ISO 26262 визначає функціональну безпеку електричних та електронних систем у транспортних засобах і вважається стандартом для функціональної безпеки автомобіля.

Першим кроком у процесі розробки ISO 26262 є визначення небезпек: аналіз ризиків та оцінка ризиків (HARA - Hazard

Analysis and Risk Assessment). Цей процес відбувається у два етапи:

– аналіз, які небезпеки можуть виникнути

– оцінка ризиків цих небезпек.

Під час аналізу ризиків розглядаються три області:

– **ступінь тяжкості** небезпеки, наприклад, чи є небезпека життю людей

– **вплив** небезпеки або наскільки імовірною є небезпека

– **керованість** небезпеки, чи здатен водій запобігти небезпеці шляхом, наприклад, гальмування.

Разом ці області визначають рівень цілісності безпеки автомобіля (ASIL - Automotive Safety Integrity Level) для цієї специфічної загрози. У цьому стандарті визначені рівні цілісності безпеки (SIL), що забезпечує нормативний метод оцінки безпеки програмованих електронних систем.

7. Методологія моделювання загроз

В цій роботі використовується методологія моделювання загроз[2], розроблена саме для автомобілів. Вона сфокусована на виявлені загроз за умови, що зловмисник вже отримав доступ до системи або можливість здійснити неправомірні дії.

Моделювання загроз складається з трьох етапів:

Етап 0. Визначення критичного програмного забезпечення (ПЗ) / систем

Для всіх ідентифікованих програм та систем:

Етап 1. Декомпозиція ПЗ / системи

1.1 Створення діаграм взаємозв'язків в транспортному засобі

1.2 Створення високорівневих потоків даних в діаграмі взаємозв’язків

Етап 2. Виявлення та аналіз загроз

2.1 Ідентифікація загрози за допомогою моделі класифікації загроз STRIDE

2.2 Визначення тяжкості загроз

8. Класифікація загроз STRIDE

Класифікація загроз STRIDE[5] була розроблена корпорацією Microsoft і використовувалась як частина їх концепції життєвий цикл безпечної розробки для класифікації та виявлення потенційних загроз. Це акронім для наступних шести категорій загроз:

1. Підроблення ідентичності
2. Порушення даних
3. Відмова від обов’язків
4. Розкриття інформації
5. Відмова в обслуговуванні
6. Перевищення привілеїв

Ідея методології STRIDE полягає в тому, щоб надати експертам із питань безпеки або тим, хто не є спеціалістом з питань безпеки інструменти для аналізу загроз безпеці.

9. Використання методології моделювання загроз

Етап 0: Визначення критичного ПЗ та систем.

Програми, які вважаються критичними, швидше за все, призведуть до серйозних загроз і мають бути досліджені в першу чергу. Цей етап можна пропустити, якщо в будь-якому випадку аналізуються всі програми або системи.

В межах методології критична програма або система - це функціональність, яка будучи скомпрометованою, може призвести до серйозних наслідків, пов'язаних з безпекою або іншими способами.

Етап 1: Декомпозиція ПЗ та системи

Метою етапу 1 є отримання повного огляду програми, системи та її компонентів. З цією метою декомпозиція програм та системи відбувається у два кроки: по-перше, створюється діаграма взаємозв'язків, що містить наступні компоненти:

• *всі компоненти, підсистеми та шини даних*: визначаються усі компоненти, які підключені до розглянутої системи;

• *зовнішні з'єднання*: виявлення будь-яких зовнішніх з'єднань, таких як WiFi, OBD-II, Bluetooth або стільниковий зв’язок;

• *типи даних*: визначення рівнів безпеки для даних у компонентах та в шинах.

Особлива увага приділяється системам та шинам, де присутні механізми зміни рівня безпеки, наприклад, шлюзи.

Етап 2. Визначення та аналіз загроз. Використовуючи діаграми з попереднього етапу, загрози ідентифікуються та

аналізуються. Це також робиться в два кроки.

Крок 1 - Ідентифікація загрози за допомогою STRIDE.

Під час першого кроку всі загрози ідентифікуються за допомогою методу моделювання загроз STRIDE. Це означає, що для кожного ЕКУ, потоку даних та зовнішнього об'єкта використовуються відповідні категорії STRIDE для ідентифікації можливих загроз, в результаті чого з'являється список загроз для всіх компонентів системи, де кожен клас STRIDE застосовується для кожного елемента.

На другому кроці кожна загроза з попереднього кроку оцінюється за двома критеріями: строгістю та керованістю. Для строгості розрізняють чотири різних сфери застосування: безпека, експлуатація, приватність та фінансова складова. Кожна сфера застосування вимірюється п’ятьма рівнями, де рівень 0 відповідає найменшому впливу, а рівень 5 – максимальному впливу на безпеку автомобіля або водія.

Визначення того, до якого класу належить кожна загроза, зазвичай робиться групою експертів, щоб обговорити деякі наслідки та покращити аналіз.

Щоб визначити ступінь тяжкості цих класів, слід використовувати класифікацію [посилання] як керівництво, що представлено в таблиці 1. Ця класифікація дає чітке розмежування між наслідками для одного або декількох транспортних засобів.

Згідно стандарту ISO 26262 рівні керованості загрозливих ситуацій такі:

– рівень 0 – повністю контрольована ситуація;

– рівень 1 – легко контролюється водієм;

– рівень 2 – помірно контролюється (більшість водіїв можуть впоратися);

– рівень 3 – водію важко контролювати ситуацію;

– рівень 4 – не може бути контрольованою.

Слід зауважити, що керованість має відношення лише до безпеки, оскільки інші категорії не контролюються водієм повністю або частково.

Використовуючи значення строгості та контрольованості загальний рівень загрози вимірюється наступним чином:

$$T = (w_s S * w_c C) + w_o O + w_p P + w_f F$$

де T – кількісна оцінка загрози, S – оцінка безпеки, O – оцінка експлуатації, P – оцінка приватності, F – оцінка фінансових ризиків, C – оцінка керованості, а також відповідні вагові коефіцієнти, що обираються експертом.

10. Висновки

Створення моделі загроз є первинним кроком для виправлення проблем безпеки, оскільки дає повний та структурований перелік вразливих систем та ПЗ системи, які необхідно захистити. Використання методології моделювання загроз дозволило дати кількісну оцінку кожній загрозі та ЕКУ автомобіля. Цей результат виявляє компоненти, які мають найбільший вплив на безпеку водія та/або автомобіля і дозволяють виробнику транспортного засобу зосередити ресурси саме на них. Тоді як результат, який надається групою експертів ґрунтується лише на їх досвіді, знаннях та перевагах, що не може вважатися об’єктивною оцінкою.

Табл. 1 – відповідність між елементами діаграм та класами STRIDE

Елемент	Підроблення ідентичності	Порушення даних	Відмова від обов’язків	Розкриття інформації	Відмова в обслуговуванні	Перевищення привілеїв
ЕКУ	✔	✔	✔	✔	✔	✔
Потік даних		✔		✔	✔	
Зовнішній елемент	✔		✔			

Список літератури

1. [Електронний ресурс] https://www.owasp.org/index.php/Application_Threat_Modeling
2. Threat Modeling for Future Vehicles, On Identifying and Analysing Threats for Future Autonomous and Connected Vehicles, Stijn van Winsen.
3. [Електронний ресурс] https://en.wikipedia.org/wiki/Electronic_control_unit
4. Comprehensive Experimental Analyses of Automotive Attack Surfaces, Stephen Checkoway, Damon McCoy, Brian Kantor, Danny Anderson, Hovav Shacham, and Stefan Savage University of California, San Diego; Karl Koscher, Alexei Czeskis, Franziska Roesner, and Tadayoshi Kohno, University of Washington.
5. [Електронний ресурс] https://msdn.microsoft.com/en-us/library/ee823878(v=cs.20).aspx

УДК 004.94

ОШИЙКО Я.Р.

ОБГРУНТУВАННЯ ПРОГРАМНОЇ АРХІТЕКТУРИ ДЛЯ СИСТЕМИ OCTOGIN

В даній статті обґрунтовано вибір програмної архітектури на стороні клієнту та серверу, розглянуто переваги та недоліки різних варіантів архітектури та приведено особливості технологічного стеку для системи Octogin.

In this article, the choice of software architecture on the client and server side is substantiated, advantages and disadvantages of different architectural options are considered, and the features of the technological stack for the system Octogin are presented.

1. Вступ

Система Octogin – це маркетинговий застосунок для блокування нав'язливої реклами та показу її за вподобаннями користувача. Її метою є покращення таргетингу реклами, що збільшить відгук користувача на неї. Для досягнення цього відбувається збір інформації про перебування користувача в мережі Інтернет, класифікації та тегування переглянутої реклами, пошук релевантних варіантів та відображення їх користувачу. Процес вибору архітектури відіграє одне з ключових значень в подальшому зборі інформації та масштабуванню системи.

2. Особливості процесу обробки даних

Процес обробки даних розділений на три частини: збір даних на основі переглянутого користувачем контенту, класифікація та тегування реклами з урахуванням контенту, представлення рекламних рекомендацій користувачу. Для класифікації та тегування реклами використано нейронні мережі та NLP (Natular-language processing). Для цього створюється модель, яка аналізує контент переглянутих користувачем веб-сторінок, отримує ключові слова і на основі них формує рекомендації по рекламі для конкретного користувача. При проектуванні архітектури та вибору технологічного стеку для вирішення цієї задачі потрібно враховувати такі критерії:

- Збір інформації не повинен впливати на роботу користувача.
- Система повинна швидко масштабуватися, оскільки відбувається постійне збільшення оброблюваної інформації та збільшується час її обробки.

3. Розрахунок навантаженості системи

На етапі вибору архітектури для системи важливим етапом є аналіз її завантаженості при різній кількості користувачів. Розрахуємо кількість запитів в секунду до сервера за кількістю підключених користувачів в годину, яка визначається таким чином:

$$H = \mathrm{S} * \mathrm{R} * \frac{\mathrm{Y}}{\mathrm{W}}$$

де S – кількість кроків користувача при одній ітерації, R – кількість запитів до серверу при одній ітерації, Y – кількість користувачів за годину, W – тривалість тесту в секундах. Теоретично взявши кількість кроків за 2 (перегляд веб-сторінки та перегляд реклами), кількість запитів за 2 (відправлення даних про переглянуту веб-сторінку та отримання релевантної реклами) та тривалість тесту в одну годину (3600 секунд) було розраховано кількість запитів в секунду для одночасних користувачів від п'яти тисяч до мільйона. Результати розрахунків відображено на рис. 1.

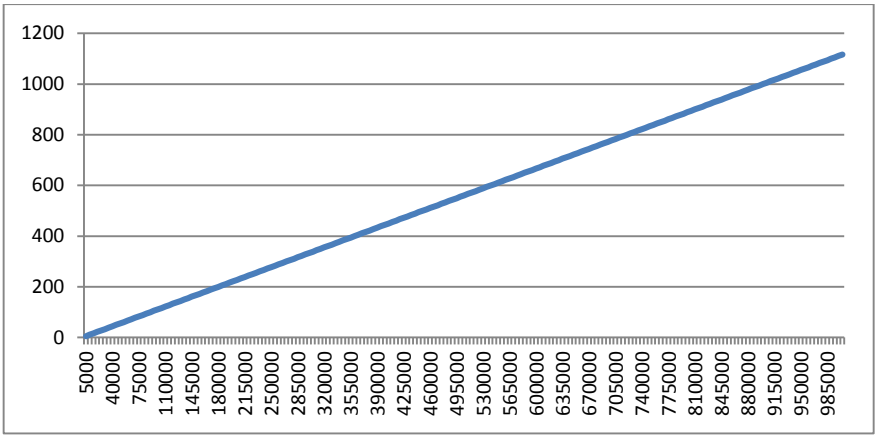


Рис. 1. Кількість запитів в секунду для різної кількості одночасних сесій користувачів

4. Вибір архітектури для системи

Типовий веб-застосунок складається з трьох складових: веб-клієнта, локального серверу та сховища даних. В системі Octogin звичайний веб-клієнт не вирішує питання збору даних, оскільки аналізується інформація інших переглянутих сторінок. Тому було вирішено розділити клієнтську частину на дві складові: веб-клієнт для ідентифікації та налаштування базових вподобань користувача та Google Chrome Extension, який відповідає за збір даних по поточній переглянутій сторінці, видалення нав'язливої реклами та заміну її релевантною користувачу.

Основним критерієм у виборі архітектури для серверної частини була легкість у масштабуванні обчислювальних потужностей та швидкого збільшення сховища даних без великих витрат в часовому та економічному аспекті. Для системи Octogin розгортання локального сервера та бази даних виявилося неефективним в плані масштабування, тому було вирішено використати хмарну платформу Firebase, яка надає документо-орієнтовану NoSQL базу даних Firestore, Firebase Cloud Functions для запуску серверного коду та безкоштовний хостинг Firebase Hosting.

Розглянемо переваги цього підходу. В системі Octogin зберігається велика кількість неструктурованої інформації, тому використання NoSQL бази даних є доцільним, оскільки не накладає обмеження на типи збережених даних. Також використання Firestore допомогло збільшити швидкодію обробки запитів, оскільки запис та отримання даних відбувається по протоколу WebSocket.

Використання Firebase Cloud Functions дозволило не розгортати локального серверу, а тримати обробку даних у хмарі. Також перевагою цього архітектурного рішення є легкість у масштабуванні, оскільки підключення додаткових потужностей відбувається автоматично.

Також на безкоштовному хостингу Firebase Hosting розташований веб-клієнт, що дозволило мінімізувати витрати.

5. Висновки

Використання хмарної архітектури дозволило мінімізувати витрати на розгортання локального серверу та вирішити основну задачу – легкість у масштабуванні. Також використання Google Chrome Extension дозволило збирати інформацію в фоновому режимі та змінювати контент веб-сторінок, вставляючи релевантну рекламу.

Список літератури

1. Leon Shklar, Richard Rosen. Web Application Architecture: Principles, Protocols and Practices. – Wiley, 2004. – 372 с
2. Martin Kleppmann. Designing Data-Intensive Applications: The Big Ideas Behind Reliable, Scalable, and Maintainable Systems 1st Edition – O'Reilly Media, 2017. – 614 с.
3. Firebase [Електронний ресурс]. – Режим доступа: https://firebase.google.com/

Оргкомітет конференції

Голова організаційного комітету: О.А. Павлов – декан факультету інформатики та обчислювальної техніки, д.т.н., професор.

Заст. голови організаційного комітету: О.М. Долголенко.

Співголови організаційного комітету:

С.Г. Стіренко – завідувач кафедри ОТ, д.т.н., професор

І.П. Муха, в.о. зав. кафедри АСОІУ, к.т.н., доцент

Члени організаційного комітету:

П.І.П.	Посада
В.М. Томашевський	Професор кафедри АСОІУ
І.В. Стеценко	Професор кафедри АСОІУ
І.В. Баклан	Доцент кафедри АСОІУ
О.Г. Жданова	Доцент кафедри АСОІУ
О.В. Гавриленко	Доцент кафедри АСОІУ
К.І. Ліщук	Доцент кафедри АСОІУ
Ю.О. Олійник	Старший викладач кафедри АСОІУ
М.О. Сперкач	Старший викладач кафедри АСОІУ
О.А. Халус	Старший викладач кафедри АСОІУ

Секретарі конференції:

С. П. Якуніна	Інженер кафедри АСОІУ
І. М. Муравйова	Старший лаборант кафедри АСОІУ

Наукова конференція студентів, магістрантів та аспірантів «Інформатика та обчислювальна техніка – ІОТ-2018». Секція кафедри автоматизованих систем обробки інформації і управління. Матеріали конференції. – Київ. – 2018. – 23-24 квітня 2018р. – 206 с.

У збірник включені тези доповідей, які були представлені на конференції “Інформатика та обчислювальна техніка» – ІОТ-2017” в секції кафедри автоматизованих систем обробки інформації і управління. В доповідях розглянуті наукові та методичні питання щодо сучасних аспектів інформатики та обчислювальної техніки.

Редакційна колегія:

Гавриленко О.В., к.ф.-м.н., кафедра АСОІУ НТУУ «КПІ ім. Ігоря Сікорського»
Олійник Ю.О., кафедра АСОІУ НТУУ «КПІ ім. Ігоря Сікорського»