1. Am Jang, Zhiwei Zou, Lang Kug Lee, Chong H. Ahn, Paul L. Bishop State-of-the-art lab chip sensors for environmental water monitoring // Meas. Sci. Technol. – 2011. – № 22. – С. 1–16. 2. Лобур М., Матвійків О., Файтас О., Методи спектроскопії та обробка даних спектрального аналізу // Вісник Національного університету "Львівська політехніка" "Комп'ютерні системи проектування. Теорія і практика". – 2011. – № 711. – С. 3–9. 3. S. Bargiel, A. Górecka-Drzazga, J. A. Dziuban, P. Prokaryn, M. Chudy, A. Dybko, Z. Brzózka, Nanoliter detectors for flow systems // Sens. Actuators. – 2004. – № 115. – С. 245–251 4. ГОСТ 2874-82 "Вода питьевая – Гигиенические требования и контроль за качеством".

УДК 519.16

Р. Базилевич, Б. Кузь
Національний університет "Львівська політехніка"
кафедра програмного забезпечення

ОПТИМІЗАЦІЯ РОЗВ'ЯЗКУ ЗАДАЧІ КОМІВОЯЖЕРА МЕТОДОМ ПАРНИХ ЗАМІЩЕНЬ

Досліджено алгоритм для оптимізації розв'язання задачі комівояжера. Зменшення довжини шляху забезпечується обміном ребер, які відповідають умові оптимізації.

Ключові слова: задача комівояжера, комбінаторна оптимізація, NP-важкі задачі.

The algorithm for TSP solution optimization is investigated. Tour minimization is performed by swapping of edges, which satisfy optimization criteria.

Key words: traveling salesman problem, combinatorial optimization, NP-hard problems.

Вступ

Пошук розв'язку задачі комівояжера вимагає значних обчислювальних затрат. Час, необхідний для якісного розв'язування задачі, стрімко зростає зі збільшенням її розмірності. Обчислювальна складність симетричної задачі становить $O((n-1)!/2)$. За допомогою жадібних та простих евристичних алгоритмів можна швидко знайти розв'язок невисокої якості навіть для задач великої розмірності. Для покращення розв'язку в таких випадках доцільно застосувати оптимізаційні алгоритми, які дають змогу зменшити довжину шляху з невеликими обчислювальними затратами. Покращення здійснюється заміною частин наявного шляху на коротші. До відомих алгоритмів розв'язування задачі комівояжера належать 2-opt та 3-opt. Оптимізація відбувається вилученням двох або більше ребер шляху та додаванням нових ребер так, щоб його довжина зменшилась. Доцільне послідовне застосування декількох оптимізаційних алгоритмів. Деякі з таких алгоритмів описано в [1].

Опис алгоритму

Для проведення оптимізації необхідна наявність деякого початкового розв'язку задачі. Такий розв'язок можна отримати одним з відомих алгоритмів, який не потребує великих затрат часу, зокрема жадібними. Початковим етапом дослідженого алгоритму є пошук потенційних для обміну множин з двох пар ребер шляху, які можна замінити коротшими. В запропонованому підході для ідентифікації таких пар використовується тріангуляція Делоне [3]. Це дає змогу зменшити пошук пар ребер, доцільних для заміни. Для побудови тріангуляції Делоне на заданій множині точок використовується одна з ефективних реалізацій алгоритму.

Алгоритм оптимізації складається з таких етапів:

1. Пошук на тріангуляції Делоне множини всіх потенційних для обміну пар ребер, після заміни яких може зменшитись довжина шляху.

2. Виділення множини пар ребер, обмін яких дає погіршення довжини шляху на величину, що не перевищує найбільшого можливого покращення.

3. Заміна всіх фрагментів, які не входять у вибрані множини, на фіксовані ребра.

4. Розв'язування задачі для утвореної множини точок.

5. Якщо довжина шляху є меншою, то заміна фіксованих ребер частинами шляху початкового розв'язку (фрагментів п.3).

Пошук потенційних для зменшення довжини шляху пар ребер

Першим етапом оптимізаційного алгоритму є пошук множини всіх пар ребер, можлива заміна яких призведе до зменшення загальної довжини шляху. Кожна пара складається з чотирьох ребер. На рис. 1 зображено приклад такої заміни. Ребра *E1*, *E2*, *E3*, *E4* можна замінити ребрами *E1'*, *E2'*, *E3'*, *E4'* відповідно.

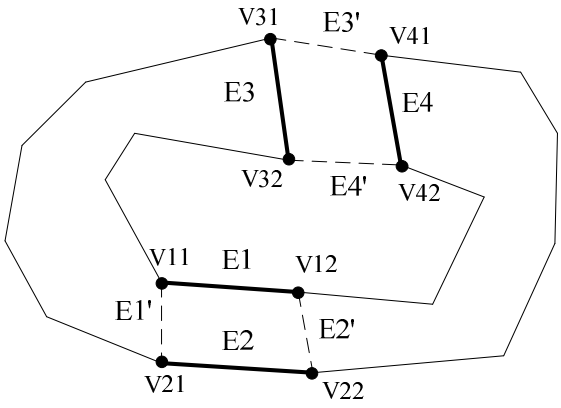

Рис. 1. Множина з двох пар ребер для оптимізації

Параметром алгоритму Δ є дозволена різниця в сумарній довжині шляху пар ребер до і після можливої заміни. Для всіх ребер виконується пошук множини пар ребер *K*, які можуть утворити групу, що зменшує довжину шляху. Початкові дані – це ребро *E1*, яке аналізується (рис. 2). Ребра *E31*, *E32*, *E33*, *E34*, *E41*, *E42*, *E43*, *E44* належать тріангуляції Делоне. Наступні кроки:

1. Пошук найкоротшого ребра *E2* серед множини ребер *E21*, *E22*, *E23*, *E24*, для якого

$$|len(E1) + len(E2) - len(E3) - len(E4)| > \Delta,$$

де пара ребер *E3* та *E4* – елемент множини [*E31*, *E41*], [*E32*, *E42*], [*E33*, *E43*], [*E34*, *E44*], якими будуть замінені ребра пари, утвореної ребром *E1* та одним з ребер *E21*, *E22*, *E23*, *E24*.

2. Додавання пар [*E1*, *E2*] та [*E3*, *E4*] до множини *K*.

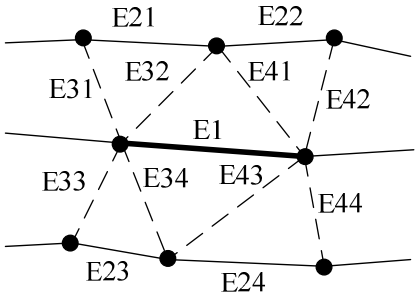

Рис. 2. Пошук можливої пари (ребро E2)
для поточного ребра (E1)

Результатом пошуку є множина *K*, в яку входять пари ребер, можлива заміна яких зменшує загальну довжину шляху. Не всі елементи з утвореної множини можна використати на наступних етапах. Кожна пара ребер для оптимізації повинна відповідати таким критеріям:

1. Для кожної вершини з множини існує шлях, який з'єднує її з вершиною іншої пари ребер.
2. Утворений шлях формує один цикл.

Введення цих критеріїв запобігає утворенню окремих циклів (рис. 3) та забезпечує пришвидшення обчислень за рахунок відкидання неякісних кандидатів. Ребра, зображені на рис. 3, непридатні для оптимізації. У разі помилкової заміни ребер *E1*, *E2*, *E3*, *E4* на *E1'*, *E2'*, *E3'*, *E4'* відповідно утворяться три цикли.

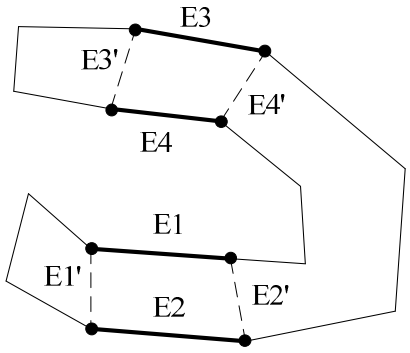

Рис. 3. Приклад порушення вимог до множини ребер для оптимізації

Всі множини з пар ребер, які не відповідають зазначеним вище критеріям, видаляються з множини *K*. Важливою особливістю є пошук ребер для заміни тільки з множини ребер тріангуляції Делоне. Така вимога забезпечує зменшення кількості можливих замін.

Оптимізація розв'язку

Множина ребер, отримана на попередніх етапах, використовується для оптимізації розв'язку. Для подальших обчислень застосовується алгоритм кільця [7]. Всі ділянки шляху, що не входять у вибрану множину ребер для оптимізації, з'єднуються фіксованими ребрами. Для розв'язування задачі методом "кільця" в експериментах використовується алгоритм LKH [5]. Після отримання розв'язку фіксовані ребера замінюють їх реальними ділянками. Тривалість виконання цього етапу залежить від розміру ділянки оптимізації.

Таблиця 1

Результати досліджень

Тест	Початкова довжина	Максимальна різниця в довжині, Δ	Кількість точок при оптимізації	Довжина після оптимізації	Час, с	Зменшення довжини, %
mona-lisa100K	6346002	5	10723	6341348	223	0.07339
vangogh120K	7227710	5	12820	7220634	352	0.09800
venus140K	7509797	3	10596	7506502	220	0.04390
pareja160K	8421635	4	12586	8417249	339	0.05211
courbet180K	8722844	3	10827	8719808	237	0.03482
usa115475	7188400	25	10848	7176318	221	0.16836
E1M.0	816704241	125	11134	816564029	199	0.01717
E316K	460113587	500	10529	459773455	176	0.07398
E100k.0	258686950	2500	13173	257678793	342	0.39125
E100k.1	259347347	2500	13280	258358766	348	0.38264

Експериментальні результати

Виконано експериментальні дослідження для тестових задач з бібліотек TSP Art [6], National TSPs [6] та DIMACS [4] на комп'ютері з процесором Phenom II X4 960T з тактовою частотою 3.0 ГГц. Розмірність тестових задач – у межах 100 000 – 1 000 000 точок. Початковий шлях утво-

рено з використанням алгоритму Quick Boruvka з системи Concorde [2]. За незначний обчислювальний час (тривалість обчислень 176 – 352 с) досягнуто покращення на 0.01717 – 0.39125 %. Максимально допустиму різницю між довжиною пар ребер до і після оптимізації вибрано з розрахунком на включення 10 000 – 12 000 точок у "кільце".

Висновки

Застосування розробленого алгоритму дає змогу покращити якість розв'язку для задачі комівояжера. Обміном двома парами ребер чи більшою кількістю можливо зменшити довжину шляху, де оптимізація в локальній області не може вплинути на заміну фрагментів, які розміщені в різних частинах робочої зони і не входять в зону локальної оптимізації.

1. Bazylevych R., Kutelmakh R., Prasad B., Bazylevych L. Decomposition and Scanning Optimization algorithms for TSP // Proceedings of the International Conference on Theoretical and Mathematical Foundations of Computer Science, Orlando, 2008, pp. 110-116. 2. Concorde: http://www.tsp.gatech.edu/concorde.html. 3. Delaunay B. Sur la sphère vide // Izvestia Akademii Nauk SSSR, Otdelenie Matematicheskikh i Estestvennykh Nauk, No 7, 1934, pp. 793–800. 4. DIMACS TSP Challenge Results: http://www.akira.ruc.dk/~keld/research/LKH/DIMACS_results.html. 5. Helsgaun K. An Effective Implementation of k-Opt Moves for the Lin–Kernighan TSP Heuristic // Datalogiske Skrifter, Writings on Computer Science, No. 109, Roskilde University, 2006. 6. TSP Art Instances: http://www.tsp.gatech.edu/data/art/index.html. 7. Базилевич Р. П., Кутельмах Р. К., Кузь Б. О. Алгоритм розв'язання задачі комівояжера великої розмірності методом "тора" // Вісник Нац. ун-ту "Львівська політехніка". – 2010. – № 686. – С. 179–182.

УДК 811.161.2

О. Дмитраш, А. Романюк, П. Тимощук
Національний університет "Львівська політехніка",
кафедра систем автоматизованого проектування

АНОТУВАННЯ ТЕКСТІВ ДЛЯ ЗДІЙСНЕННЯ АВТОМАТИЧНОГО РОЗПІЗНАВАННЯ ІМЕНОВАНИХ СУТНОСТЕЙ

Описано створення анотованого корпусу іменованих сутностей для української мови.

Ключові слова: видобування інформації, розпізнавання іменованих сутностей, корпус.

This paper describes the process of creating an annotated corpus of named entities for Ukrainian language.

Key words: information extraction, named entity recognition, corpus.

Постановка проблеми

Сьогодні, у період постійного зростання обсягів інформації, все гострішою стає необхідність створення ефективних підходів і систем опрацювання неструктурованої інформації та перетворення їх на структуровані дані. Створення та функціонування таких систем безпосередньо залежить від систем видобування інформації.

Видобування інформації – це одне із завдань обробки природної мови, яке полягає в автоматичному видобуванні структурованих даних із неструктурованих або напівструктурованих інформаційних джерел. В останні роки системи видобування інформації набули широкої