УДК 519.8 doi:10.15421/321616

В.А. Турчина, Є.В. Ротгауз
Дніпропетровський національний університет імені Олеся Гончара

МІНІМІЗАЦІЯ СЕРЕДНЬОЗВАЖЕНОГО ЧАСУ У ЗАДАЧАХ УПОРЯДКУВАННЯ

Розглянуто розроблення наближених методів побудови упорядкування для узагальненої задачі паралельного упорядкування. Наведено детальний опис методів та вказівки щодо їх реалізації. Представлено результати аналізу розроблених методів на основі отриманих тестових результатів.

Рассмотрено разработку приближенных методов построения упорядочения для обобщённой задачи параллельного упорядочения. Приведено подробное описание методов и указания относительно их реализации. Представлено результаты анализа разработанных методов на основе полученных тестовых результатов.

This publication explores the area of parallel sequencing through graph theory. Parallel sequencing is a generalization of extreme ordering problem, mathematical model of which is based on dependency graphs. Research includes formulation of two different generalized problems of parallel ordering. Two approximation algorithms, based on modified flow algorithms, were developed during the research. Evaluation of the objective function of a sequence was prepared as a result of this research.

Ключові слова: задачі теорії розкладів, оптимізаційні задачі на графах, задача паралельного упорядкування, методи пошуку максимального потоку мінімальної вартості.

Вступ

Задачі оптимального розподілу ресурсів є узагальнення екстремальних задач упорядкування. Проблема упорядкування полягає в знаходженні такої послідовності виконання операцій на кожному об'єкті технологічної або виробничої системи, яка виключає його невиправдані простої. Методи оптимального упорядкування мають широкий спектр застосування. Їх застосовують у разі розробки гнучких автоматизованих виробництв, систем штучного інтелекту і програмних комплексів планування обчислювальних процесів. Ряд прикладних задач, таких як моделювання кліматичної системи, ядерні дослідження, дослідження літальних апаратів, часто потребують розв'язання задачі в режимі реального часу. До таких задач, зокрема, відносять і задачі розпаралелення обчислень для ЕОМ. Вони пов'язані з оптимальним розміщенням завдань, на порядок яких накладають технологічні обмеження. Оптимізації підлягають різні величини. В даній роботі розглянемо задачі, які потребують мінімізації середнього зваженого часу знаходження задач у системі.

Незважаючи на удавану простоту, більшість із цих задач є NP-важкі, тому точні методи їх розв'язання, які мають експоненційну складність, становлять перш за все теоретичний інтерес[2].

У даний час роботи в галузі розв'язання практичних задач присвячені вивченню, розробці та використанню наближених алгоритмів поліноміальної складності.

Постановка задачі

У класичній постановці задача має таке формулювання [4]:

Розглянемо задачу упорядкування, критерієм вартості в якій візьмемо зважену суму часів закінчення всіх завдань або середньозважений час їх завершення.

Складниками побудови упорядкування є:

– система завдань *V* – індексована множина з *n* завдань, $n > 0$;
– залежність завдань: у *класичній постановці* завдання незалежні, тобто відношення часткового порядку $\prec = \emptyset$;
– додатне число τ_j для кожного *j* з *V* означає потребу завдання *j* в часі обслуговування;
– дійсне число c_j називають вартістю перебування завдання *j* в системі. В детермінованій розглядуваній задачі всі вартості знаходження завдання в системі дорівнюють 1.

Випадок 1. Розглядаючи задачу для одного процесора, можна обмежитися розкладом з перестановкою завдань. Перестановка $s = s_1,...,s_n$ елементів $I(F)$ сумісна з частковим порядком, тобто якщо

$$j_{s_k} \prec j_{s_k'}, \text{ то } s_k < s_k'.$$

Середньозважений час завершення для розкладу, визначеного перестановкою *s,* обчислюватимемо як значення виразу:

$$mwft(s) = \sum_{j=1}^{n} c_{s\,j} \left(\sum_{i=1}^{j} \tau_{s\,i}\right).$$

Перестановку s, яка в даному випадку є впорядкування, назвемо оптимальною, якщо $mwft(s)$ досягає мінімуму серед усіх перестановок.

Випадок 2. Розглянемо детерміновану задачу упорядкування для декількох процесорів. Кількість процесорів дорівнює m.

Задачу опишемо матрицею часів виконання завдань. Нехай $[\tau_{ij}]$ – матриця розміром $m*n$, де τ_{ij} – час виконання завдання j на процесорі i для $1 \leq i \leq m$ та $1 \leq j \leq n$.

Обмеження на незалежність завдань унеможливлює розв’язання більшої кількості реальних задач. Тому сформулюємо узагальнену задачу. Розглянемо узагальнену детерміновану задачу упорядкування для декількох процесорів. Уведемо три умови:

– завдання залежні, тобто $\prec \neq \emptyset$;
– всі вартості знаходження завдання в системі дорівнюють 1;
– залежності між завданнями слід задавати за допомогою графа залежностей.

Відтак розглянемо розв’язання узагальненої задачі паралельного упорядкування шляхом зведення до сіткової задачі та подальшого знаходження оптимального розкладу з використанням методу пошуку оптимального потоку мінімальної вартості. Такий підхід є виправданий за умови, що вартість знаходження задач у системі дорівнює одиниці. Узагальнена задача зводиться до розв’язання k класичних задач за допомогою зведення графа залежностей до зручного вигляду та його подальшого дослідження.

Зведення до задачі пошуку оптимального потоку

Нехай маємо упорядкування S, згідно з яким завдання j виконуватимемо на i-тому процесорі та є k завдань – на цьому ж процесорі за завданням j. Тоді величина τ_{ij} робить внесок часу до всіх k завдань. Виділимо окремо члени, що залежать від τ_{ij}.

Скориставшись отриманими коефіцієнтами при τ_{ij}, створюємо матрицю внесків задач у загальний час виконання розкладу C. Переформулюємо задачу як задачу пошуку потоку мінімальної вартості. На основі матриці C формуємо $(S\text{-}t)$ мережу, застосовуючи C як матрицю вартостей у мережі. Елемент $c(i,j)$ визначає вартість переносу одиниці потоку з точки x_i до точки y_j. Потік називатимемо оптимальним, якщо його величина дорівнює n, а вартість *cost* [5] –

$$cost(f) = \sum_{i=1} \sum_{j=1} c(x_i, y_j) f(x_i, y_j) \to \min$$

по всіх допустимих потоках величини *G*. Знайшовши оптимальний потік мінімальної вартості, отримаємо розв'язок задачі упорядкування, згідно з яким для кожного завдання визначимо процесор та місце в упорядкуванні на ньому.

Метод пошуку оптимального потоку

Методи пошуку екстремального потоку засновані на послідовній побудові екстремальних потоків $f_0, ..., f_p$ таких, що величина $f_p = p$. Допустимий потік *f* у мережі *G* називатимемо екстремальним, якщо він має мінімальну вартість серед усіх допустимих потоків. Початковий потік f_0 має нульові значення на всіх дугах *G*. Нехай ми отримали f_p для деякого $p<n$. Для знаходження f_{p+1} виділимо шлях з x_0 до y_0, уздовж якого потік може бути збільшений на 1, при цьому вартість одиничного збільшення є мінімальна серед вартостей усіх шляхів. Такий шлях називатимемо збільшуючий шлях мінімальної вартості. Алгоритм завершується, коли в залишковій мережі не існує шляхів від *s* до *t*. Тоді потік є максимальний і такий, що відповідає допустимому розв'язку проблеми мінімізації вартості потоку [3].

Для пошуку найкоротшого збільшуючого шляху мінімальної вартості застосовано алгоритм **Successive ShortestPathAlgorithm**, який шукає максимальний потік та одночасно оптимізує цільову функцію *cost*.

Розв'язання узагальненої задачі упорядкування. Алгоритм Stagedmethod

Узагальнену задачу розв'язуватимемо шляхом зведення її до певної кількості класичних задач. Проведемо дослідження графа залежностей, який задає відношення часткового порядку між поданими n завданнями. Для графа застосовували упорядкування спеціального вигляду *S*. Вершини, що входять до множини кожного рівня упорядкування *S[r]*, є незалежні для кожного *r*. Тому для всіх рівнів існує задача упорядкування, задана матрицею $[\tau_{ij}]_r$, для якої виконується зведення до сіткової задачі та пошук оптимального потоку. Фінальне упорядкування S^* формуємо за допомогою послідовного зіставлення точних упорядкувань S_r кожного рівня. Під час зіставлення упорядкувань S_r у фінальному упорядкуванні з'являється новий елемент – очікування *W*. Його вартість знаходження в системі дорівнює 0, але цей елемент впливає на функцію *mwft* завдань, виконуваних після нього. Даний алгоритм будує *наближений* розв'язок через наявність очікувань під час операції послідовного склеювання множини точних розв'язків.

Отже, цільову функцію наближеного алгоритму визначаємо таким чином:

$$mwft(S^*) = \sum_{i=1}^{m} \sum_{j=1}^{|S[i]|} \sum_{k=1}^{j} (\tau_{ij_k} + w_{ij_k}),$$

де k – місце завдання τ_{ij} або w_{ij} в упорядкуванні на i-тому процесорі, $|S[i]|$ – кількість елементів перестановки на i-тому процесорі.

Недолік алгоритму полягає в тому, що під час зіставлення отриманого упорядкування S^* можливі випадки, коли з'являється невиправдано велика кількість простоїв процесора через очікування між кожним рівнем розкладів. Для вирішення цієї проблеми запропоновано алгоритм мінімізації часу простою процесора на основі методу гілок та меж.

Оптимізація наближеного алгоритму побудови упорядкування для узагальненої задачі багатопроцесорних систем. Алгоритм Stagedmethodwithpackingoptimization

Розв'язуємо наступну задачу: з множини «завдань» треба обрати підмножину так, щоб максимізувати сумарну наповненість інтервалу очікування, при цьому не перевищуючи обмеження на максимально припустимий час заповнення.

Для цього скористаємося методом гілок та меж. Алгоритм методу буде працювати для кожного інтервалу очікування на кожному процесорі окремо. Для нерозподілених задач формуємо транспортну мережу та розв'язуємо класичну задачу.

Оскільки необхідно максимізувати наповненість інтервалу, то оцінкою зверху виступає довжина інтервалу очікування процесору. Для кожного завдання враховуватимемо додаткові очікування через залежність до інших задач, якщо вони наявні. На *k-му* кроці алгоритму враховуємо час виконання завдань, розташованих в упорядкуванні перед поточним. За підмножини розбиття допустимої множини слугуватимуть усі незалежні вершини, для яких час виконання відповідних завдань не перевищує час проміжку очікування.

Для даної задачі скористаємося додатковою оцінкою цільової функції. Оскільки необхідно мінімізувати середній час знаходження завдань в системі, то застосуємо для неї оцінку знизу. Необхідність оцінювання як ще один рівень оптимізації виникає лише за умови рівності оцінки наповненості інтервалу очікування в різних вузлах дерева.

Такий підхід є наближений і не приводить до оптимального результату через додаткові очікування, які з'являються за наявності часткового порядку між завданнями, а втім це зумовлює значне зменшення часу простоїв процесорів.

Аналіз результатів

Обидва обрані алгоритми тестувались для задачі побудови розкладів на сукупності тестів з різними заданими кількостями залежних робіт в кожному (від 5 до 1000 робіт).

У процесі побудови розкладу алгоритми проходили такі кроки.

1. Ініціалізація вхідних даних: матриці часів виконання завдань на даних процесорах [τ_{ij}] та граф залежностіG.
2. Застосування методу побудови розкладу по рівнях **Stagedmethod**.
   - 2.1. Для алгоритму Stagedmethod будується оптимальне упорядкування для кожного з трьох рівнів окремо алгоритмом **Successive ShortestPathAlgorithm.**
   - 2.2. Склеювання отриманих упорядкувань за часом виконання найдовшого завдання з кожного рівня.
   - 2.3. За отриманим розкладом розраховується функція *mwft.*
3. Застосування методу побудови розкладу з покроковою оптимізацією по рівнях **Stagedmethodwithpackingoptimization.**
   - 3.1. Будується оптимальне упорядкування для завдань першого рівня аналогічно упорядкуванню за **Stagedmethod.**
   - 3.2. Визначається наявність очікувань та незалежних завдань, які можуть очікування заповнити.
   - 3.3. Для виявлених незалежних завдань працює метод гілок та меж.
   - 3.4. Для нерозподілених завдань будується мережа і визначається оптимальний процесор.

Результати тестів показали, що алгоритм **Stagedmethodwithpacking optimization** має не гірші, а у більшості випадків кращі показники: оптимальні середній зважений час знаходження завдань в системі та час побудови розкладу.

Наприклад, для розрідженого графа залежності, визначеного для 40 завдань, **Stagedmethodwithpackingoptimization** дає більш оптимальні результати, ніж **Stagedmethod**. В розріджених графах з'являється велика кількість очікувань, які легко заповнити за допомогою методу гілок та меж, оскільки залежність між вершинами послаблена.

Результат. Оптимізований метод дає результат на ***-111*** одиниць часу менше за звичайний метод склеювання розкладів. Маємо виграш ***+18%.*** За рахунок зменшення кількості завдань, які потребують роботи з транспортною мережею, алгоритм **Stagedmethodwithpackingoptimization** значно швидше обробляє дані.

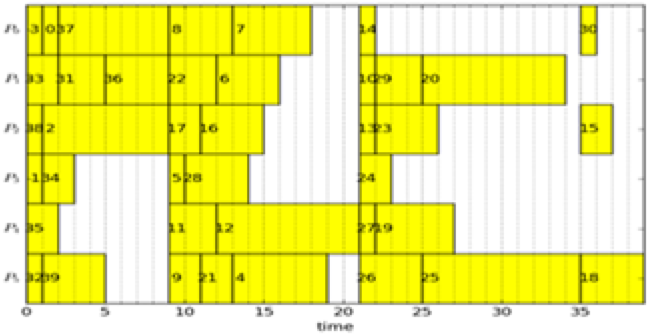

Рис 1. Результат роботи методу Staged method

Staged method (took 654.109 ms to finish):
Total flow time = 630

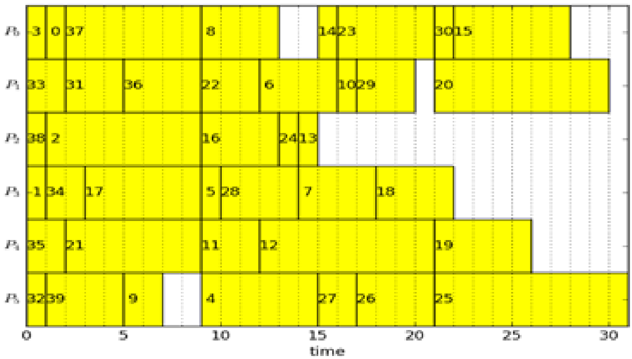

Рис.2. Результат роботи методу Staged method with packing

Staged method with packing optimization (took 199.326 ms to finish):
Total flow time = 519

Висновок

У даній роботі сформульована узагальнена задача мінімізації критерію середньозваженого часу. Для розв’язання поставленої задачі запропоновано та протестовано два нові наближені методи побудови упорядкування на основі пошуку потоку мінімальної вартості в сітковій мережі.

Запропоновані наближені методи є більш ефективні за точні методи послідовного перебору з погляду часової складності, та уможливлюють розв’язання практичних задач з кількістю робіт порядку тисячі за прийнятний час.

Бібліографічні посилання

1. **Воеводин, В.В.** Параллельные вычисления [Текст] / В.В. Воеводин, Вл.В. Воеводин. – СПб.: БХВ-Петербург, 2002. – С.210–240.
2. **Hall, L.A.** Approximation Algorithms for NP-hard problems [Text] / L.A. Hall // PWS Publishing Company, 1997. – № 1. – P. 972–983.
3. **Кормен, Т.** Алгоритмы. Построение и анализ [Текст] / Т. Кормен, Ч. Лейзерсон, Р. Ривест. – М.: «И.Д. Вильямс», 2013. – С.747–788.
4. **Коффман, Э.Г.** Введение в теорию расписаний [Текст] / Э.Г. Коффман. – М., «НАУКА», 1984. – С. 77–78.
5. **Ahuja, R. K.** Network Flows: Theory, Algorithms, and Applications [Text] / R. K. Ahuja, T. L. Magnanti, J. B. Orlin. – Prentice Hall, 1993. – № 1. – P. 145–170.
6. **Бурдюк, В.Я.** Алгоритмы параллельного упорядочения [Текст]: учеб. пособие / В.Я. Бурдюк , В.А. Турчина. – Д., 1985. – 80 с.

Надійшла до редколегії 15.04.2016