Моделиране на клас паралелни задачи чрез минимално оцветяване на граф

Ивайло Пенев

Modeling of a class of parallel jobs using the graph colouring problem: *The paper is concerned with a class of jobs which are common about programming systems in the financial calculations domain and particularly in the portfolio management systems. The jobs perform evaluation of various financial objects. As the evaluations of the objects are independent, parallel execution of the jobs is possible. For effective realization preliminary scheduling of the parallel execution is necessary, satisfying conditions, the most important of which is preventing the concurrent access of the jobs to common data source. In accordance with the limiting condition an objective function is defined. The minimal value of the function guarantees least time for conclusion of all jobs. A proof, that the scheduling problem described is NP-hard is presented. Reduction of the problem to the well known graph colouring problem is used. Proving the NP-hardness is significant for designing of quick algorithm for solving the problem.*

Key words: *parallel jobs, scheduling, graph colouring, NP-hardness.*

УВОД

В статията са разгледани клас паралелни задачи с общ (конкурентен) достъп до общ източник на данни. Такива задачи се формират в изчислителни системи, обслужващи различни области, като особено характерни са в системите за управление на финансови портфейли. В тези системи една задача изпълнява оценяване на финансов портфейл. По същество оценяването представлява изпълнение на множество от анализи върху позициите на портфейла. Тези позиции се описват чрез исторически данни, които се съхраняват в общ източник (най-често база от данни). Съществуват условия за паралелна реализация на оценките на всички портфейли чрез използване на изчислителните ресурси, съществуващи във всяка съвременна организация. Необходимо е разработване на подход за преодоляване на едновременния достъп до общия източник, който ограничава постигането на ефективно паралелно изпълнение [7]. В много съществуващи системи оценките за финансовите обекти са реализирани като командни (batch) процедури, предназначени за изпълнение в неработно време. В тези случаи използването на традиционни програмни интерфейси и библиотеки като MPI, PThreads, OpenMP не е оправдано поради необходимост от сложни и скъпи преобразования на програмния код.

Възможен подход за решаване на този проблем е въвеждане на „задържане" при стартирането на дадена задача с цел освобождаване на общия ресурс, използван от други изпълняващи се задачи. Необходимо е предварително определяне на сценарий, при който подходящо задържане на някои от задачите осигурява минимален конкурентен достъп до общия източник. За тази цел може да се използват предсказани времена на задачите, основани на съществуващата зависимост между позициите на даден портфейл и времето за оценка. Това предсказване би могло да се реализира например чрез регресионен анализ на извадка от примерни портфейли и измерване на времената за оценка [2].

ФОРМАЛНО ОПИСАНИЕ И КЛАСИФИКАЦИЯ В ТЕОРИЯТА НА ПЛАНИРАНЕ

Може да се направи следното формално описание на проблема за планиране на паралелното изпълнение на разглеждания клас задачи в разпределена изчислителна среда.

Множество от паралелни задачи $J_1,J_2,\ldots,J_N$ се изпълнява в разпределена изчислителна среда, съставена от еднакви компютри $P_1,P_2,\ldots,P_M$. Времето за изпълнение на задача се състои от време за четене на данни T_{READ}, време за

изпълнение на изчислителни операции τ_{CALC} и време за запис на резултати τ_{STORE} в общ източник на данни. Времената са предварително известни.

Общото време за паралелно изпълнение на всички задачи в средата се увеличава със стойност τ_{DELAY}, която е забавяне, получено в следствие на конкурентния достъп на задачите до общия източник. Целта е да се направи такова планиране на изпълнението на всички задачи в изчислителната среда, при което да бъдат удовлетворени следните две условия:

1. Свеждане на времето τ_{DELAY} до минимум.

2. Постигане на минимално общо време за изпълнение на всички задачи, т.е. $\min\left(\sum_{i=1}^{n}\tau_{READi}+\tau_{CALCi}+\tau_{STOREi}\right)$.

Задачата за планиране на изпълнението на множество от задачи в дадена изчислителна среда се описва с наредена тройка *α* | *β* | *γ* , където трите параметъра имат съответно следното значение [8]:

- брой на изчислителните възли;

Среда, съставена от *m* еднакви компютри се означава *Pm*.

- ограничителни условия – условия, които трябва да са удовлетворени при изпълнението;

Например, когато е недопустимо прекъсване на изпълнението на стартирана задача, то за параметъра *β* не се задава стойност;

- целева функция (оптимизационен критерий);

Избраният алгоритъм за планиране трябва да осигурява минимална (или максимална) стойност на целевата функция. Поради това целевата функция се разглежда като критерий за оптималност при избор на алгоритъм. Най-често използваните целеви функции, описани в литературата са време за приключване на последната (т.е. „най-дългата") задача (обикновено се бележи с C_{max}) и общо време за завършване на всички задачи ($\sum C_i$) [4, 8].

Освен класификация на задачи теорията описва т.нар. правила за избор на алгоритъм за планиране. Тези правила и класовете от задачи, за които са предназначени са подробно описани например в [8].

За някои класове задачи е доказано, че са NP-трудни (NP-hard). Класифицирането на задача като NP-пълна (NP-complete) или NP-трудна (NP-hard) означава, че не съществува бърз алгоритъм за решението и. За решението на такива задачи най-често се прилагат евристични подходи.

При планирането на задачите в описания клас системи е необходимо да се предотврати (или минимизира) конкурентния достъп на задачите до общ източник на данни. Това налага дефиниране на нова целева функция, свързана с броя на времевите интервали, в които две или повече задачи осъществяват едновременен достъп до източник на данни по време на четене или съхраняване на резултати. Следователно проблемът за планиране в описания клас задачи се описва като P_m || *f*, където целевата функция *f* е брой времеви интервали, в които две или повече задачи осъществяват едновременен достъп до общия ресурс (източник на данни). Дефинирането на такава функция (критерий) изисква проектиране на алгоритъм, осигуряващ планиране, при което задачите осъществяват минимален конкурентен достъп до общия ресурс.

В този смисъл описаният клас задачи може да се опише като Pm || $\sum\tau_{DELAY}$, където използваните символи имат следното значение:

Pm - изчислителна среда, съставена от *m* еднакви изчислителни възли,

|| - не се допуска прекъсване на изпълнението на стартирана задача,

$\sum\tau_{DELAY}$ - целева функция, която се определя като брой на интервалите, в които при конкретен сценарий на планирането две или повече задачи осъществяват едновременен достъп до общия източник на данни.

За да бъде проектиран алгоритъм за бързо решаване, от важно значение е да се определи дали задачата е NP-трудна. Известни са множество задачи, за които е доказано, че са NP-пълни. В първата енциклопедия на NP-пълните задачи [6] е представено доказателство за NP-трудност на голям брой известни задачи.

Много често използван подход за доказване на NP-трудност е свеждане (редуциране) на изследваната задача до някоя от известните NP-трудни задачи. Този подход намира широко приложение в теорията на планиране за доказване на NP-трудност на някои класове задачи за планиране. В [8] е доказана NP-трудност на следните класове задачи за планиране чрез редуциране до известни NP-трудни задачи:

- Намиране на минимално време за приключване на най-дълга задача в среда от два еднакви компютъра чрез свеждане до задачата за разделяне на множество от цели числа (Partition problem). Това свеждане се означава кратко като P2||Cmax ∞ Partition problem.
- Намиране на минимално време за приключване на най-дълга задача в т.нар. *Job shop* среда [8] при зададена възможност за прекъсване на задачи по време на изпълнение чрез свеждане до задачата за разделяне на множество от цели числа на три множества (3-partition problem) или J2|rcrc,prmp| ∞ 3-partition.
- Намиране на минимално време за приключване на най-дълга задача в среда от един компютър със зададен интервал от време между две задачи чрез свеждане до задача за намиране на Хамилтонов цикъл в граф (Hamiltonian cycle), т.е. 1 $|s_{jk}|$ Cmax ∞ Hamiltonian**.**

Тези резултати водят до идеята, че проблемът за планиране на паралелно изпълнение на дефинирания клас задачи в среда от *m* еднакви изчислителни възли може да се сведе до някоя от известните задачи, с което да се покаже нейната NP-трудност.

МОДЕЛИРАНЕ НА РАЗГЛЕЖДАНИЯ КЛАС ЗАДАЧИ ЧРЕЗ МИНИМАЛНО ОЦВЕТЯВАНЕ НА ГРАФ

Времето за изпълнение на всяка задача се разделя на времеви интервали, напр. всяка секунда (или милисекунда) от изпълнението представлява интервал. Тогава задачата за планиране се свежда до намиране на сценарий, при който броят на интервалите с конкурентен достъп до общия източник е минимален.

За да бъде изяснена идеята за представяне на паралелното изпълнение чрез граф, ще разгледаме опростен случай с две задачи, които изпълняват само четене на данни и изчислителни операции със следните примерни времена, изразени в брой времеви единици (интервали):

Задача J_1: $\tau_{READ1} = 1, \tau_{CALC1} = 1$. Изпълнението на задачата във времеви интервали може да бъде представено чрез следната диаграма:

R_1	C_1

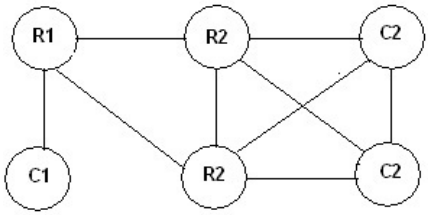
, където R – read (четене на данни за периодите на анализирания инструмент), C – calculate (симулационни изчисления).

Задача J_2: $\tau_{READ2} = 2, \tau_{CALC2} = 2$. Съответно представяне във времеви интервали:

R_2	R_2	C_2	C_2

Идеята е да се отложи стартирането на една от задачите за толкова на брой времеви интервали колкото са необходими на другата задача за достъп (т.е. четене

или запис на данни) до общия източник. Следователно е необходимо да се симулират сценарии с различно отлагане (отместване) на стартирането на задачите. В разглеждания опростен случай с две задачи паралелното изпълнение може да се представи чрез следния неориентиран граф.



Фиг. 1. Представяне на паралелно изпълнение на две задачи чрез граф

Върховете на графа представят етапите на всяка задача, разделени по времеви интервали. С R_1 и C_1 са означени съответно интервалите на четене и симулационни изчисления на първата задача. Върховете R_2, R_2, C_2, C_2 представят по два последователни интервала на четене на данни и симулационни изчисления за втората задача. Дъгите свързват върхове, обозначаващи онези времеви интервали, в които задачите не могат (или не трябва) да се изпълняват едновременно. Дъгите R_1R_2 обозначават едновременен достъп на двете задачи до общия източник при четене на данни. Дъгите R_1C_1, R_2C_2 представят последователното изпълнение на четене на данни и изчислителни операции за всяка задача.

Това представяне позволява свеждане на проблема за планиране на разглеждания клас задачи до известната задача за оцветяване на граф (graph coloring problem). Тази задача и нейни приложения са разгледани в много източници, например в [1, 3].

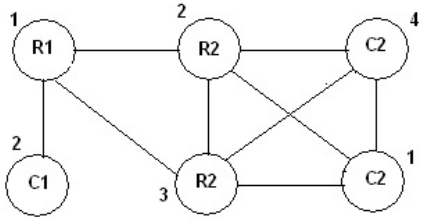
Задачата за оцветяване на граф се състои в съпоставяне на цвят от множество на всеки връх на даден граф така, че никои два съседни върха да не бъдат оцветени с еднакъв цвят. Оцветяването на върхове разделя (разбива) множеството от върхове на графа на подмножества, всяко от които съдържа върхове, оцветени с еднакъв цвят. Тези подмножества се явяват независими поради отсъствие на съседни върхове в едно подмножество.

Тази задача е една от най-известните NP-трудни задачи. Строго формално доказателство за NP-трудност е представено например в [5].

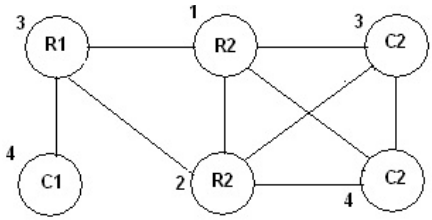
Следователно, ако се покаже, че проблемът за планиране на разглеждания клас задачи може да се представи чрез минимално оцветяване на върховете на граф, това означава, че дефинираният клас задачи са също NP-трудни.

В разглеждания пример оцветяването на граф съответства на планиране на изпълнението на задачите по времевата ос, при което изкуствено се въвежда отложено стартиране на задача (или задачи в случай на повече от две задачи). Всеки отделен цвят съответства на времеви интервал, в който задача изпълнява обозначената операция (в случая четене на данни или изчисляване).

Следващите фигури представят някои възможни оцветявания:



Фиг. 2. Вариант 1 за минимално оцветяване



Фиг. 3. Вариант 2

Оцветяването в първия вариант (фиг. 2) показва, че паралелното изпълнение на двете задачи може да се реализира в четири времеви интервала без интервали на конкурентен достъп до общия източник. Означаването на върха C2 с цвят 1 прави този сценарий на планиране невъзможен. Прекъсване на изпълнението на задача не е допустимо, поради което двата времеви интервала C2 трябва да бъдат последователни. Трябва да се търси сценарий, при който върховете, обозначаващи етапите на изпълнение на задача да бъдат означени с цветове с последователни номера.

Вторият вариант (фиг. 3) на минимално оцветяване представя планиране на изпълнението, при което етапите на всяка задача са в последователни интервали. Първата задача се изпълнява във времеви интервали 3 и 4, т.е. стартирането на задачата 1 е отложено с два интервала спрямо стартирането на задача 2. Този сценарий може да бъде представен във времеви интервали по следния начин:

		R_1	C_1
R_2	R_2	C_2	C_2

Вижда се, че отложеното стартиране предотвратява конкурентния достъп на задачите до общия източник и общото време за приключване на двете задачи е минимално – четири времеви интервала.

Следователно задачата за планиране на паралелно изпълнение на описания клас задачи, което да предотвратява конкурентния достъп на задачите до общ ресурс (източник на данни) и в същото време да осигурява минимално общо време за изпълнение на всички задачи е NP-трудна (NP-hard) задача. Може да се обобщи, че:

Pm || $\Sigma \tau_{DELAY}$ е NP-трудна задача.

ЗАКЛЮЧЕНИЕ

Класифицирането на задача като NP-трудна очертава важни насоки за нейното решаване. Известно е, че за намиране на алгоритъм за решаване на NP-трудна задача трябва да бъде приложен някой от следните подходи [1]:

- използване на евристичен алгоритъм – т.е. по-бърз метод, който да дава решение само в част от случаите;
- приблизително решаване на задачата – за някои NP-трудни задачи съществуват алгоритми за решаване с приближение. Те не решават проблема точно, но винаги намират решение, за което може да се докаже, че е достатъчно близко до точното.

Освен това за решаване на задачата за минимално оцветяване на граф са известни алгоритми [3], които могат да бъдат приложени и при планирането на разгледания клас задачи.

Представеният подход за моделиране чрез граф може да намери приложение при планиране на изпълнението на паралелни задачи в области, където има необходимост от обработване на големи обеми от данни – статистика, медицина, икономика.

Реалните системи за управление на финансови портфейли изискват изпълнение на голям брой задачи. Това води до образуване на графи с голям брой върхове и дъги, за оцветяването на които са известни различни алгоритми. Съществуват както алгоритми за точно решение, така и алгоритми с приближение. Бъдещата работа е свързана с изследване на известни алгоритми за оцветяване на граф при планиране на описания клас задачи.

Също така ще бъде проектиран, реализиран и тестван алгоритъм за оцветяване на граф, съобразен със спецификите на разглеждания клас задачи, а именно конкурентен достъп до общ източник на данни. Алгоритъмът ще оцветява върховете на подграфа на всяка задача с цветове с последователни номера. По този начин планирането ще задава предписание за изпълнението на всяка задача по времеви интервали.

Благодарности

Тази разработка е финансово подкрепена от проект: "Подкрепа на творческото развитие на докторанти, пост-докторанти и млади учени в областта на компютърните науки", BG051PO001-3.3.04/13".

ЛИТЕРАТУРА

[1] Наков П., Добриков П., Програмиране =++ Програмиране, стр. 332, 356, Top Team Co., 2002.

[2] Пенев Ивайло, Модел за предсказване на времената за изпълнение при оценяване на клас финансови обекти, сп. „Компютърни науки и технологии”, брой 1/2010, година VIII, стр. 20-23.

[3] Christofides N., Graph Theory – an Algorithmic Approach, Academic Press, 1975.

[4] Dutot Pierre-Francois, G. Mounie, D. Trystram, Scheduling Parallel Tasks Approximation Algorithms, Handbook of Scheduling: Algorithms, Models and Performance Analysis, edited by Joseph Y-T. Leung, CRC Press, Boca Reaton, FL, USA, 2004.

[5] Erickson J., Algorithms, Lecture notes, University of Illinois, 2011, http://theory.cs.uiuc.edu/~jeffe/teaching/algorithms/everything.pdf.

[6] Garey M., D. Johnson, Computers and intractability: A Guide to the Theory of NP-Completeness, Freeman, 1979.

[7] Penev I., A. Antonov, Realization of Portfolio Management System in a distributed computing environment, International Conference Computer Science. Sofia, 2009.

[8] Pinedo M.. *Scheduling: Theory, Algorithms, and Systems*. ISBN: 978-0-387-78934-7. Springer Science+Business Media, LLC. New York, 2008.

За контакти:

Ас. инж. Ивайло Пенев, катедра “Компютърни науки и технологии”, Технически Университет - Варна, e-mail: ivailopenev@yahoo.com

Докладът е рецензиран.