УДК 519.7, 519.8

Г. Г. Цегелик, Р. П. Краснюк
Львівський національний університет імені Івана Франка

ЗАДАЧА ОПТИМАЛЬНОГО РОЗПОДІЛУ ЗАВДАНЬ МІЖ КОМП'ЮТЕРАМИ МЕРЕЖІ

З використанням методу динамічного програмування розроблено ефективний обчислювальний алгоритм оптимального розподілу завдань між комп'ютерами мережі та розв'язано числовий приклад, який демонструє ефективність запропонованого алгоритму.

Ключові слова: оптимальний розподіл, комп'ютерна мережа, методи динамічного програмування, обчислювальні алгоритми.

The efficient computational algorithm for optimal allocation of tasks between computers on the network has been developed. It has been designed by using dynamic programming method. The numerical example that demonstrates the effectiveness of the proposed algorithm has been solved.

Key words: optimal distribution, computer network, dynamic programming methods, computation algorithms.

Вступ

Проблему зростання потреб в обчислювальних ресурсах та одночасне прагнення скоротити витрати на обладнання можна вирішити завдяки впровадженню *GRID*-технологій, як базової технології розподілених обчислень, у процес побудови обчислювальних систем. Замість значних витрат коштів на суперкомп'ютер *GRID*-технології дозволяють використати потужності вже наявних комп'ютерів у мережі організації, оскільки середнє завантаження їхніх процесорів, як правило, не перевищує декількох відсотків. Особливо це важливо для підприємств, що використовують великі інформаційно-обчислювальні ресурси, які динамічно виділяються для розв'язання громіздких задач, в науковій, індустріальній, адміністративній та комерційній сферах діяльності. Застосування технологій *GRID* дозволяє будувати систему управління розподіленими обчислювальними ресурсами. У такій ситуації користувачеві вже не важливо, на якому конкретному вузлі мережі розв'язується його задача: він просто споживає певну кількість віртуальної процесорної потужності, наявної в мережі. *GRID*-системи гармонійно доповнюють низку обчислювальних архітектур, що використовуються сьогодні. У *GRID* час взаємодії між вузлами вимірюється мілісекундами й секундами, тому такі системи не призначені для розв'язання паралельних задач, а націлені здебільшого на виконання пакетних завдань, коли кожне окреме завдання виконується на одному вузлі [1].

Питанням моделювання різних аспектів функціонування розподілених систем, зокрема побудованих на *GRID*-технології, та проблемам системної інтеграції присвячено роботи таких вітчизняних та зарубіжних учених: М. Я. Бартіша, В. В. Воєводіна, М. З. Згуровського, В. М. Коваля, О. Я. Матова, О. А. Павлова, О. В. Палагіна, С. Ю. Яковлева, Г. Г. Цегелика, I. Foster, L. Fusco, D. Menasce, L. Hluchy, B. Javadi, R. Buyya та ін.

З обчислювального погляду *GRID*-системи за відповідної організації обчислювального процесу дають змогу за прийнятний час загалом розв'язати два класи задач: задачі, в яких можливе паралельне виконання обчислювального процесу, або потоку задач, для яких розпаралелювання неможливе. Можливий варіант одночасного обслуговування *GRID*-системою двох визначених вище типів задач. Тому при управлінні *GRID*-системою постає завдання організації обчислень, що забезпечує їх оптимальний режим. Як наслідок, побудова та дослідження оптимізаційних моделей, пов'язаних з функціонуванням комп'ютерних мереж, є актуальною та важливою проблемою, яка розглядається у роботі.

Математична постановка проблеми.

Нехай необхідно розподілити m завдань однакової складності серед n комп'ютерів $K_1, K_2, \ldots, K_n$ мережі (комп'ютери можуть мати не однакову потужність). Відомо час виконання завдань на кожному комп'ютері. Нехай $t_i(x_j)$, $i = 1,2,\ldots,n$, − час виконання $x_j = j$, $j = 1,2,\ldots,m$, завдань на i-му комп'ютері K_i. Завдання полягає в такому розподілі завдань серед комп'ютерів, щоб сумарний час виконання завдань був мінімальним.

Якщо позначити через x_i, $i = 1,2,\ldots,n$ кількість завдань, що планується виконати на i-му комп'ютері, то математична модель матиме вигляд:

$$T = \sum_{i=1}^{n} t_i\left(x_i\right) \to \min \tag{1}$$

за умов

$$\sum_{i=1}^{n} x_i = m, \quad x_i \in \{1,2,\ldots,m\}, i = 1,2,\ldots,n. \tag{2}$$

Для розв'язування поставленої задачі (1)–(2) використаємо метод динамічного програмування [2].

Формулювання обчислювального алгоритму

Позначимо через $T_i(x_j)$, $i = 1,2,\ldots,n$, − сумарний час виконання $x_j = j$, $j = 0,1,\ldots,m$, завдань на перших i комп'ютерах $K_1, K_2, \ldots, K_i$, а через $T_i^*(x_j)$, $i = 1,2,\ldots,n$, − сумарний час виконання завдань у разі оптимального розподілу $x_j = j$, $j = 0,1,\ldots,m$, завдань серед перших i комп'ютерів $K_1, K_2, \ldots, K_i$. Виконання завдання розіб'ємо на n кроків.

На першому кроці визначаємо мінімальний час у разі виконання $x_j = j$, $j = 0,1,2,\ldots,m$, завдань на першому комп'ютері K_1. На другому кроці визначаємо мінімальний час у випадку виконання $x_j = j$, $j = 0,1,2,\ldots,m$, завдань на перших двох комп'ютерах K_1, K_2. І т.д. Нарешті, на n-му кроці визначаємо мінімальний час у разі виконання m завдань на n комп'ютерах $K_1, K_2, \ldots, K_n$. На першому кроці приймемо $T_i\left(x_j\right) = t_i\left(x_j\right)$, $T_i^*\left(x_j\right) = t_i\left(x_j\right)$, $j = 0,1,\ldots,m$.

На другому кроці:

$$T_2\left(x_j\right) = \begin{cases} t_2(0) + T_1^*\left(x_j - 0\right), \\ t_2(1) + T_1^*\left(x_j - 1\right), \\ \quad \ldots \\ t_2\left(x_j\right) + T_1^*(0). \end{cases} \text{ і } T_2^*\left(x_j\right) = \min_{0 \le k \le j}\left\{t_2(k) + T_1^*\left(x_j - k\right)\right\}, \text{ для } j = 0,1,\ldots,m.$$

Взагалі, на s-му кроці ($s = 3,4,\ldots,n-1$)

$$T_s\left(x_j\right) = \begin{cases} t_s(0) + T_{s-1}^*\left(x_j - 0\right), \\ t_s(1) + T_{s-1}^*\left(x_j - 1\right), \\ \quad \ldots \\ t_s\left(x_j\right) + T_{s-1}^*(0). \end{cases} \text{ і } T_s^*\left(x_j\right) = \min_{0 \le k \le j}\left\{t_s(k) + T_{s-1}^*\left(x_j - k\right)\right\}, \text{ для } j = 0,1,\ldots,m.$$

На останньому n-му кроці досить обчислити $T_n(m)$ і $T_n^*(m)$, де

$$T_n(m) = \begin{cases} t_n(0) + T_{n-1}^*(m), \\ t_n(1) + T_{n-1}^*(m-1), \\ \quad \ldots \\ t_n(m) + T_{n-1}^*(0). \end{cases} \text{ і } T_n^*(m) = \min_{0 \le k \le m}\left\{t_n(k) + T_{n-1}^*(m-k)\right\}.$$

Оптимальний розподіл m задач серед n комп'ютерів мережі визначаємо так.

Нехай $T_n^*(m)$ досягає мінімуму для $k = l_1$ тоді l_1 завдань треба виконувати на останньому n-му комп'ютері K_n. Далі необхідно розподілити m-l_1 завдань серед перших n-1 комп'ютерів $K_1, K_2, \ldots, K_{n-1}$. Припустимо, що $T^*_{n-1}(m-l_1)$ досягає мінімуму для $k = l_2$. Це означає, що l_2 завдань

треба виконувати на (n-1)-му комп'ютері K_{n-1}. Якщо $T^*_{n-2}(m-(l_1+l_2))$ досягає мінімуму для $k = l_3$, то l_3 завдань необхідно розв'язувати на (n-2)-му комп'ютері K_{n-2}. І т.д. Нехай $T^*_2(m-(l_1+l_2+\ldots+l_{n-2}))$ досягає мінімуму для $k = l_{n-1}$. Тоді l_{n-1} завдань треба виконувати на другому комп'ютері K_2. Нарешті, $l_n = m-(l_1+l_2+\ldots+l_{n-1})$ завдань треба виконувати на першому комп'ютері K_1.

Мінімальний сумарний час виконання всіх завдань становитиме $T_n^*(m)$ одиниць.

Числовий приклад реалізації алгоритму

Для демонстрації алгоритму розглянемо тестовий приклад. Необхідно розподілити вісім завдань однакової складності серед трьох комп'ютерів мережі. Час виконання завдань кожним комп'ютером залежно від їх кількості подано у табл. 1.

Таблиця 1

Час виконання завдань комп'ютерами мережі

$t_i(x_j)$	x_j								
	0	1	2	3	4	5	6	7	8
$t_1(x_j)$	0	5	7	9	11	14	17	21	25
$t_2(x_j)$	0	3	6	9	14	17	21	25	29
$t_3(x_j)$	0	4	5	6	8	11	15	19	24

На першому кроці алгоритму приймемо $T_1(j) = T_1^*(j) = t_1(j),\ j = 0,1,\ldots, 8.$

На другому кроці визначаємо: $T_2(0) = T_2^*(0) = t_2(0) + T_1^*(0);$

$$T_2(1) = \begin{cases} t_2(1) + T_1^*(0), \\ t_2(0) + T_1^*(1), \end{cases} \quad T_2^*(1) = \min_{0\le k\le 1}\left\{t_2(k) + T_1^*(1-k)\right\};$$

$$T_2(2) = \begin{cases} t_2(2) + T_1^*(0), \\ t_2(1) + T_1^*(1), \\ t_2(0) + T_1^*(2), \end{cases} \quad T_2^*(2) = \min_{0\le k\le 2}\left\{t_2(k) + T_1^*(2-k)\right\};$$

$$T_2(3) = \begin{cases} t_2(3) + T_1^*(0), \\ t_2(2) + T_1^*(1), \\ t_2(1) + T_1^*(2), \\ t_2(0) + T_1^*(3), \end{cases} \quad T_2^*(3) = \min_{0\le k\le 3}\left\{t_2(k) + T_1^*(3-k)\right\};$$

$$T_2(4) = \begin{cases} t_2(4) + T_1^*(0), \\ t_2(3) + T_1^*(1), \\ t_2(2) + T_1^*(2), \\ t_2(1) + T_1^*(3), \\ t_2(0) + T_1^*(4), \end{cases} \quad T_2^*(4) = \min_{0\le k\le 4}\left\{t_2(k) + T_1^*(4-k)\right\};$$

$$T_2(5) = \begin{cases} t_2(5) + T_1^*(0), \\ t_2(4) + T_1^*(1), \\ t_2(3) + T_1^*(2), \\ t_2(2) + T_1^*(3), \\ t_2(1) + T_1^*(4), \\ t_2(0) + T_1^*(5), \end{cases} \quad T_2^*(5) = \min_{0\le k\le 5}\left\{t_2(k) + T_1^*(5-k)\right\};$$

$$T_2(6)=\begin{cases} t_2(6)+T_1^*(0), \\ t_2(5)+T_1^*(1), \\ t_2(4)+T_1^*(2), \\ t_2(3)+T_1^*(3), \\ t_2(2)+T_1^*(4), \\ t_2(1)+T_1^*(5), \\ t_2(0)+T_1^*(6), \end{cases} \quad T_2^*(6)=\min_{0\le k\le 6}\left\{t_2(k)+T_1^*(6-k)\right\};$$

$$T_2(7)=\begin{cases} t_2(7)+T_1^*(0), \\ t_2(6)+T_1^*(1), \\ t_2(5)+T_1^*(2), \\ t_2(4)+T_1^*(3), \\ t_2(3)+T_1^*(4), \\ t_2(2)+T_1^*(5), \\ t_2(1)+T_1^*(6), \\ t_2(0)+T_1^*(7), \end{cases} \quad T_2^*(7)=\min_{0\le k\le 7}\left\{t_2(k)+T_1^*(7-k)\right\};$$

$$T_2(8)=\begin{cases} t_2(8)+T_1^*(0), \\ t_2(7)+T_1^*(1), \\ t_2(6)+T_1^*(2), \\ t_2(5)+T_1^*(3), \\ t_2(4)+T_1^*(4), \\ t_2(3)+T_1^*(5), \\ t_2(2)+T_1^*(6), \\ t_2(1)+T_1^*(7), \\ t_2(0)+T_1^*(8), \end{cases} \quad T_2^*(8)=\min_{0\le k\le 8}\left\{t_2(k)+T_1^*(8-k)\right\}.$$

Результати обчислень наведено у табл. 2.

Таблиця 2

Результати обчислень за другим кроком алгоритму

x_j	k	$t_2(k)$	$T_1^*(x_j-k)$	$T_2(x_j)$
1	2	3	4	5
0	0	0	0	0^*
1	1	3	0	3^*
	0	0	5	5
2	2	6	0	6^*
	1	3	5	8
	0	0	7	7
3	3	9	0	9^*
	2	6	5	11
	1	3	7	10
	0	0	9	9^*
4	4	14	0	14
	3	9	5	14
	2	6	7	13
	1	3	9	12
	0	0	11	11^*

Продовження табл. 2

1	2	3	4	5
5	5	17	0	17
	4	14	5	19
	3	9	7	16
	2	6	9	15
	1	3	11	14^*
	0	0	14	14^*
6	6	21	0	21
	5	17	5	22
	4	14	7	21
	3	9	9	18
	2	6	11	17^*
	1	3	14	17^*
	0	0	17	17^*
7	7	25	0	25
	6	21	5	26
	5	17	7	24
	4	14	9	23
	3	9	11	20^*
	2	6	14	20^*
	1	3	17	20^*
	0	0	21	21
8	8	29	0	29
	7	25	5	30
	6	21	7	28
	5	17	9	26
	4	14	11	25
	3	9	14	23^*
	2	6	17	23^*
	1	3	21	24
	0	0	25	25

На третьому кроці $T_3(8)=\begin{cases} t_3(8)+T_2^*(0), \\ t_3(7)+T_2^*(1), \\ t_3(6)+T_2^*(2), \\ t_3(5)+T_2^*(3), \\ t_3(4)+T_2^*(4), \\ t_3(3)+T_2^*(5), \\ t_3(2)+T_2^*(6), \\ t_3(1)+T_2^*(7), \\ t_3(0)+T_2^*(8), \end{cases}$ $T_3^*(8)=\min\limits_{0\le k\le 8}\left\{t_3(k)+T_2^*(8-k)\right\}.$

Результати обчислень наведено у табл. 3

Таблиця 3

Результати обчислень за третім кроком алгоритму

x_j	k	$t_3(k)$	$T_2^*(x_j-k)$	$T_3(x_j)$
8	8	24	0	24
	7	19	3	22
	6	15	6	21
	5	11	9	20
	4	8	11	19^*
	3	6	14	20
	2	5	17	22
	1	4	20	24
	0	0	23	23

Із табл. 3 бачимо, що $T_3^*(8)=19$ і досягається для $k = 4$. Це означає, що чотири завдання треба виконувати на третьому комп'ютері. Залишилось розподілити чотири завдання серед перших двох комп'ютерів. Із табл. 2 бачимо, що $T_2^*(4)=11$ і досягається для $k = 0$. Це означає, що на другому комп'ютері не потрібно виконувати завдань, а чотири завдання необхідно виконувати на першому комп'ютері. Загальний час виконання всіх завдань становить 19 одиниць часу.

Висновки та перспективи подальших наукових розвідок

Концепція *GRID* орієнтована на створення комп'ютерної інфраструктури нового типу, що забезпечує глобальну інтеграцію інформаційних і обчислювальних ресурсів, тобто *GRID* виступає універсальною ефективною інфраструктурою для високопродуктивних розподілених обчислень та обробки даних. Як наслідок, актуальність досліджень *GRID*-технологій та систем не викликає сумнівів, і тому в роботі було здійснено математичну постановку задачі оптимального розподілу завдань між комп'ютерами мережі з врахуванням їхніх можливостей та запропоновано ефективний обчислювальний алгоритм виконання цього завдання, що грунтується на методі динамічного програмування. Зазначимо, що вибір методу побудови розв'язання поставленої задачі визначався характером математичної моделі. Для демонстрації ефективності запропонованого алгоритму здійснено побудову розв'язання тестової задачі. Як видно з наведеної схеми розв'язання, алгоритм є простим до реалізації об'єктно-орієнтованою мовою програмування. Предметом наших подальших досліджень з цієї проблематики є інтеграція запропонованого обчислювального алгоритму в програмне забезпечення, яке здійснює керування ресурсами *GRID*-системи.

1. Менаске Д., Алмейда В. Производительность Web-служб. Анализ, оценка и планирование. – СПб.: ДиаСофт, 2003. 2. Цегелик Г.Г. Математичне програмування: навч. посібник. – Львів: ЛНУ імені Івана Франка, 2011.