УДК 004.08

С.М. Гриша, О.В. Гриша, Д.О. Шевчик

ЗАДАЧА КАЛЕНДАРНОГО ПЛАНУВАННЯ ДЛЯ GRID- ОБЧИСЛЕНЬ

Аннотація: Пропонується генетичний алгоритм для задачі календарного планування для GRID-обчислень.

Ключові слова: GRID-обчислення, генетичні алгоритми, календарне планивання.

Вступ

Ідея грід-компьютінга виникла разом з розповсюдженням персональних комп'ютерів, розвитком Інтернету і технологій пакетної передачі даних на основі оптичного волокна (SONET, SDH і ATM), а також технологій локальних мереж (Gigabit Ethernet). Смуга пропускання комунікаційних засобів стала достатньою, щоб при необхідності залучити ресурси іншого комп'ютера. Враховуючи, що безліч підключених до глобальної мережі комп'ютерів велику частину робочого часу простоює і має в своєму розпорядженні ресурси більші, ніж необхідні для вирішення їх повсякденних завдань, виникає можливість застосувати їх невживані ресурси в інтересах обчислень.

З іншого боку зростання складності та кількості прийомів комп'ютерного моделювання призвела до значного збільшення потреби у комп'ютерних потужностях. Звичайні багатопроцесорні суперкомп'ютери та, навіть, кластери не можуть задовольнити обчислювальних потреб задач моделювання з великою деталізацією, на відміну від GRID-систем, що мають чудові характеристики масштабування і можуть надавати величезні обчислювальні потужності.

Предметом дослідження є GRID-системи. За своїми особливостями виділяють три основні типи GRID-систем:

- GRID на основі використання вільного ресурсу персональних комп'ютерів, що добровільно надаються (добровільна GRID);
- наукові GRID - застосування, що розпаралелюють, програмуються спеціальним чином (наприклад, з використанням Globus Toolkit);
- GRID на основі виділення обчислювальних ресурсів на вимогу (Enterprise GRID або комерційна GRID) - звичайні комерційні застосування працюють на віртуальному комп'ютері, який, у свою чергу, складається з декількох фізичних комп'ютерів, об'єднаних за допомогою GRID-технологій.

Розглянемо основні особливості GRID-систем, що відрізняють їх від звичайних паралельних обчислювальних систем:

- паралельний комп'ютер зазвичай повністю гомогенний. У свою чергу, GRID-система може об'єднувати в собі вузли з різними типами процесорів, об'ємом пам'яті і т.д.;

 ISSN 1562-9945

- паралельний комп'ютер зазвичай має окрему, оптимізовану мережу зв'язку з високою пропускною здатністю, яка зазвичай має фіксовану топологію. У свою чергу, мережа зв'язку GRID-системи може мати високий ступінь гетерогенності та незбалансованості, включати в себе набір різних між-машинних мережевих з'єднань і різноманітних з'єднань через Інтернет, пропускна здатність і показники латентності яких можуть сильно змінюватись в процесі роботи системи;
- паралельні комп'ютери зазвичай мають постійну конфігурацію. На відмінність від них, доступність комп'ютерних ресурсів у GRID-системах непостійна і може з часом змінюватись;
- паралельний комп'ютер зазвичай керується однією операційною системою і постачає лише базовий функціонал, такий як доступ до файлової системи чи керування ресурсами. У свою чергу, GRID-система по своїй природі є системою об'єднаних одиничних обчислювальних вузлів, та об'єднує потенційно дуже різноманітні операційні системи та програмне забезпечення.

Обґрунтування задачі дослідження

Метою дослідження є оптимізація прибутку від виконання множини завдань, роботи яких враховують можливості розпаралелювання обчислень. Засобом виконання завдань є GRID-система, вузли якої мають певну вартість, коефіцієнти збільшення терміну виконання за рахунок віддаленого зв'язку та є бізнес-процес динамічного залучення та звільнення нових вузлів у GRID-систему на певний термін на комерційній основі.

Проблеми використання ресурсів для спільної обробки у GRID-системах широко досліджуються. Чонг-Єн Лі у роботах [1,2,3] запропонував модель динамічного аналізу ресурсів, яка здатна отримувати інформацію про завантаження центрального процесора, про кількість виконуваних завдань кожним вузлом GRID-системи, для того щоб досягнути рівномірного розподілення навантаження і зробити планування та розподілення ресурсів зв'язаних вузлів оптимальним. Також він запропонував алгоритм MPUMTT(максимальне завантаження процесора і мінімальний оборотний час), який досягає максимального завантаження ЦП утримуючи мінімальним оборотний час. Алгоритм повністю контролює увесь хід виконання робіт в системі і здатний ефективно розподіляти ресурси пам'яті та завантажувати процесорні ресурси. З іншого богу, алгоритм породжує велику кількість службової інформації, що впливає на навантаження мережі. До того ж, задля високої ефективності роботи алгоритму потрібен високий ступінь синхронізації виконання робіт. Костенко В.А. у роботі [4] запропонував ітераційні алгоритми для побудови та оптимізації розкладів робіт. Кращі з них дозволяють отримувати гарні результати, але на заваді впровадженню таких алгоритмів стоїть їх велика потреба в обчислювальних ресурсах та ресурсах пам'яті, яка стрімко зростає разом із збільшенням розмірів задачі.

Усі ці дослідження не пропонують рішень для оптимізації комерційного використання вільних обчислювальних ресурсів мережі у складі GRID-системи для виконання складних моделювань чи інших “важких” обчислень, поєднаних у завдання з можливістю паралельного виконання тредів робіт.

Постановка задачі

Розглядатимемо задачу побудови оптимального за вартістю динамічного розкладу виконання множини завдань обчислення на GRID-системі.

В загальному випадку кожний з вузлів GRID-системи має свою специфіку ресурсів і при залученні цих ресурсів треба враховувати найбільш релевантні цій специфіці завдання та елементи завдань. Проте, в даній постановці ми поки вважатимемо , що ресурси є відмінними лише за об’ємами оперативної, дискової пам’яті та потужністю процесорів. Вікна можливого використання нічим не пов’язані окрім ціни між собою і не можуть слідувати безпосередньо один за одним у часі. Тому кожне з вікон вузла будемо вважати окремим ресурсом.

Введемо позначення:

I – множина завдань, кожне з яких $i \in I$ має:

t_i^n – термін подачі на виконання;

c_i – вартість результату виконання;

Δ_i – штраф за несвоєчасне виконання, що є кусочно-лінійною функцією від реального строку виконання $t_i^з$. Кожний з послідовних інтервалів затримки δ_{il} додає до сумарного штрафу значення γ_{il}. Таким чином загальний штраф за затримку i- го завдання

$$\Delta_i = \sum_{l \in L_i, l \leqslant l_k} \gamma_{il}, t_i^з \in l_k, l_k \in L_i,$$

де l_k – k-й штрафний інтервал, в який потрапив строк здачі готового завдання.

J – пул ресурсів у складі GRID-системи, що може бути динамічно залучений на комерційній основі за певним графіком. Таким чином кожний з можливих вузлів – ресурсів $j \in J$ має певну кількість періодичних інтервалів можливого залучення (в термінах доби чи місяця) $h \in H_j$, за цінами $b_{jh} \in B_j$, $h \in H_j$.

Кожне завдання має певну структуру виконання та розпаралелювання алгоритму w_i. Тобто:

$$w_i =< s_i, r_i >, s_i \in S_i, r_i \subset s_i \times s_i, i \in I,$$

де s_i – множина елементів внутрішньої структури завдання $s_i \in S_i$. Для кожного елементу відомий розподіл вірогідності тривалості виконання p_{s_i};

r_i – відношення часткового порядку виконання елементів внутрішньої структури завдання, що визначається на декартовому добутку над s_i, $r_i^ф \subset s_i \times s_i$.

 ISSN 1562-9945

Для множини I, що складається з завдань певної структури з елементами, розрахованими на паралельні обчислення, згідно розкладу доступного пулу ресурсів Jзнайти розклад такий, що

$$C = \sum_{i \in I} c_i - \sum_{i \in I} \Delta_i - \sum_{i \in I} (\sum_{j \in J_i} (\sum_{h \in H_j} (x`_{ijh} - x``_{ijh})) * b_{jh}), b_{jh} \in B_j,$$
$$(x`_{ijh}, x``_{ijh}] \in x, x \in X, h \in H_j;$$
$$C- > \max,$$

де C – прибуток від виконання множини завдань;

$x`_{ijh}$ – початок h-го використання з множини використань H_j ресурсу $j \in J_i$ для i-ї роботи;

$x``_{ijh}$ – кінець h-го використання з доступного інтервалу H_j ресурсу $j \in J_i$ для i-ї роботи ;

H_j – множина термінів використань з доступного ресурсу $j \in J_i$.

Розклад є коректним, якщо кожний елемент кожного завдання призначений на ресурс і притому лише на один ресурс; частковий порядок, заданий на початковій множині робіт, не порушився в розкладі; розклад є безтупіковим.

Вирішення задачі

Оскільки доступні інтервали вузлів можна використовувати незалежно від приналежності до одного вузла, чи різних, але ідентичних по ресурсам, вважатимемо ресурсо-інтервал окремим вузлом, що може бути залучений у розклад незалежно. Також приймемо до уваги, якщо процес на зміг завершити своє виконання у доступний інтервал, він повинен бути знятий і запущений на іншому вузлі GRID-системи.

Ця задача є задачею календарного планування. Для її вирішення застосуємо генетичний алгоритм на еволюційній моделі, яка складається з наступних елементів:

- базова множина рішень X– усі можливі розклади на загальному пулі ресурсів,

- оператор побудови початкової популяції: оператор, який дозволяє виділити

на множині X його підмножини – таким оператором є імітаційна модель процесу генерації допустимих розкладів $X_n \in X, n = \{1, 2, ..., 100\}$, тобто початкова популяція складається з 100 розкладів.

- оператор кросовера $K : X \times X \to X$ діє на основі імітаційного алгоритму побудови потомка з врахуванням функціональної готовності до виконання та поточного стану елементу в батьківських розкладах. Виділення ресусів елементу розкладу в момент конкуренції за ресурс виконується за алгоритмом:

1. Якщо елемент в цей момент виконується, чи виконаний в обох батьків, йому призначається підходящий вільний ресурс;
2. Якщо елемент в цей момент не почав виконуватися для жодного з батьків, він чекає і переходимо до наступного елементу;

3. Якщо елемент в цей момент виконується, чи виконаний лише в одного з батьків, приймаємо випадкове рішення про надання ресурсу.

- оператор мутації $M : X \to X$. Мутацією є випадкова зміна послідовності надання ресурсів з врахуванням доступності по інтервалам;

- критерієм відбору вважаємо прибуток C від виконання усієї множини завдань за результатами обчислювального експерименту над імітаційною моделлю;

- оператор селекції виділяє підмножину популяції для виконання кросовера з вірогідністю

$$P(X_n) = (1 - \frac{C_n}{\max\limits_{X_n \in X}(C_n)});$$

- оператор відбору визначає батьків нового члену популяції як найбільш віддалені з селектованої множини розклади

$\sum\limits_{i \in I}(\max(x“_{1ijh}) - \max(x“_{2ijh}))^2 \to \max,$

де $j \in J_i, h \in H_j,\ (x‘_{1ijh}, x“_{1ijh}) \in x_1, (x‘_{2ijh}, x“_{2ijh}) \in x_2, x_1, \in X, x_2 \in X;$

– оператор загибелі частини популяції виконується при зростанні потужності популяції. Загибель виникає для розкладів з вірогідністю

$$P(X_n) = (\frac{C_n}{\max\limits_{X_n \in X}(C_n)});$$

- еволюція має зупинку за умови непокращення прибутку C від виконання множини завдань, чи перевищення терміну надання гарантій замовнику обчислення.

Експериментальні дослідження алгоритму

Експериментальні дослідження були проведені на програмно реалізованій імітаційній моделі. Для проведення тестових експериментів був реалізований генератор завдань з паралельними алгоритмами реалізації, що підлягають плануванню. Отримувані в результаті генерації завдання містили у своїй структурі від 3 до 10 паралельних тредів. Тестування проводилося для 20 вузлів GRID-системи з 2-3 інтервалами надання ресурсу.

При дослідженні алгоритму складання розкладу виконання тестових програм отримані наступні результати:

- 30-50 ітерацій дозволяють отримати оптимальний розклад, надалі поліпшення розкладу не відбувається або відбувається незначне поліпшення;
- якість отримуваного рішення на 10 - 20 % краще первинного.

Отримані показники справедливі для популяції розміром 100. Збільшення розміру популяції дозволяє отримати рішення за меншу кількість кроків, але потребує значних ресурсів.

Виводи

Показано, що поєднання евристичного та генетичного алгоритму з імітаційним механізмом побудови популяції забезпечує отримання ефективного рішення в умовах експоненційної складності задачі складання розкладу для GRID-системи.

Література

1. Lee H-.M., Lee T.-Y., Yang C.-H., and M.-H. Hsu "An Optimal Analyzing Resources Model Based on Grid Environment," WSEAS Transactions on Information Science and Applications, Vol.3, No.5, pp.960-964, 2006.
2. Lee H-.M., Lee T.-Y., and Hsu M.-H. "A Process Schedule Analyzing Model Based on Grid Environment ," KES 2006, Part III. LNAI 4253, Springer-Verlag, Berlin Heidelberg, pp.938-947, 2006
3. Chong-Yen Lee, Wuu-Yee Chen, and Tsang-Yean Lee "Optimization of Job Schedule Model Based on Grid Environment", Journal of Networks, Vol 3, No 3 (2008), pp. 27-33, Mar 2008
4. Костенко В.А. "Оценки сложности и качества различных итерационных алгоритмов построения рас писаний" // Фак-т ВМиК МГУ им. М.В. Ломоносова. - М. - 2004. - С. 161-167.

Отримано 09.12.2009 р.