УДК 004.632:004.75

А.О. Шемет

АНАЛІЗ ПАРАЛЕЛЬНОГО АЛГОРИТМУ БЛОЧНОГО РЕЗЕРВНОГО КОПІЮВАННЯ

Анотація. В цій статті наведено алгоритм блочного резервного копіювання даних. Виконано декомпозицію задачі резервного копіювання з використанням паралельних систем. Проаналізовано паралельний алгоритм, наведено ярусно-паралельний та інформаційний графи Розраховано мінімальну кількість процесорів, потрібну для виконання алгоритму.
Ключові слова: паралельний алгоритм, інформаційний граф, щільність завантаження, ярусно-паралельний граф, строки виконання операторів, резервне копіювання інформації.

Вступ. Розвиток сучасних інформаційних систем безпосередньо пов'язаний з накопиченням, обробкою та передаванням даних. Важливою задачею є забезпечення збереження цих даних у незмінному стані або створенні можливості отримання даних у незмінному вигляді на момент потреби. Одним з напрямків підвищення надійності збереження даних є їх програмне резервування – створення надлишковості. Методи та алгоритми послідовного програмного резервного копіювання даних були описані в роботах [1] та [2]. В рамках вирішення задачі розпаралелення алгоритмів резервного копіювання інформації в роботах [3] та [4] було модифіковано метод інкрементального резервного копіювання, запропоновано метод блочного резервування, оптимізований до використання апарату паралельних обчислень.

Постановка задачі. В даній роботі проведемо аналіз паралельного алгоритму в рамках вирішення задач знаходження оптимального плану робіт для метода паралельного блочного резервного копіювання, знаходження мінімальної кількості процесорів (вузлів), які потрібні для проведення резервного копіювання за час, що не перевищує заданий.

Основна частина. В роботі [4] приведено алгоритм блочного резервного копіювання. Алгоритм розділено на підалгоритми: сегмен-

тація даних, аналіз вмісту джерела інформації та відповідних резервних копій (пошук та порівняння блоку джерела з відповідним блоком резервного сховища), перенесення блоків даних до резервних копій, індексація сегментів у резервних сховищах, хешування блоків даних.

Для алгоритму блочного резервного копіювання приведемо граф алгоритму (у ярусно-паралельній формі).

$$G = (V, E) \qquad (1)$$

У якості вузлів графу є операції $V = \{1, \ldots, |V|\}$, а у якості дуг E – вхідні дані, або ті що є результатом виконання операцій на попередньому ярусі [6].

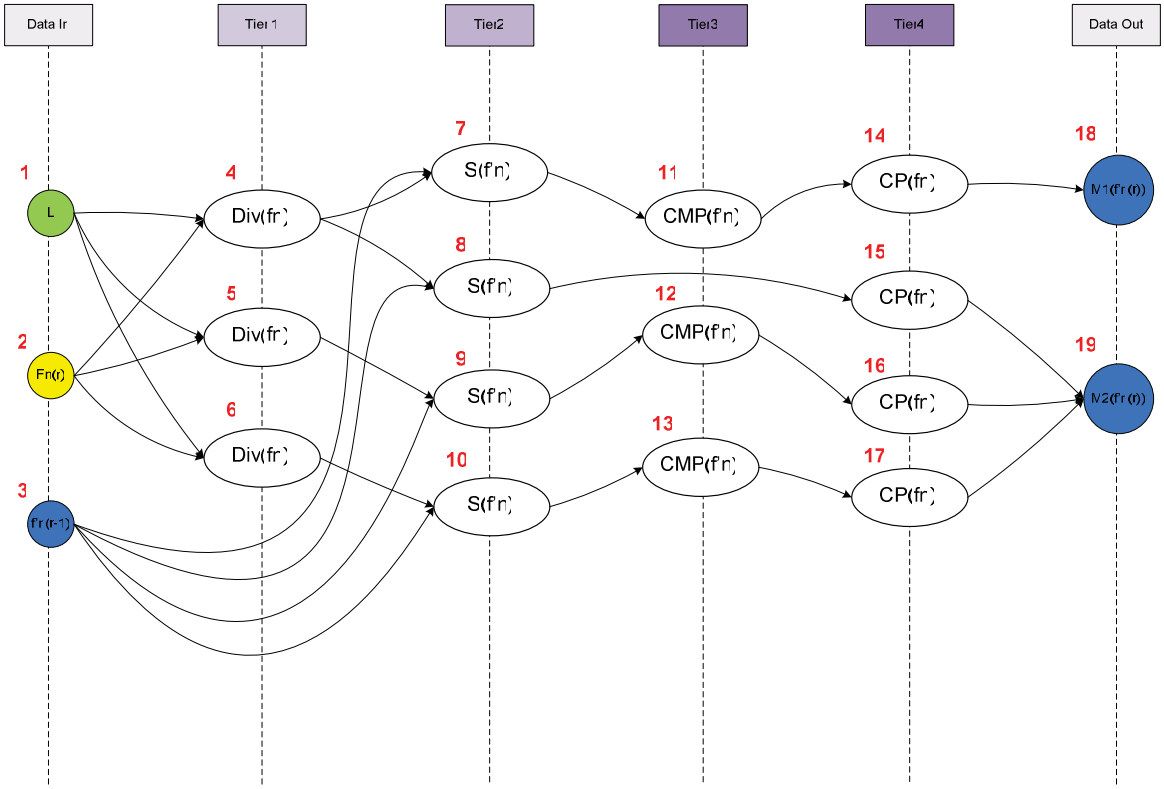

Рисунок 1 - Ярусно-паралельний граф алгоритму

Розглянемо граф алгоритму більш детально (Рис.1.). Задача резервного копіювання розділена на 4 ярусів, тобто висота ЯПФ складає 4. Ширина ЯПФ дорівнює 4. На вхід подано наступні дані: $f_n(r)$ – файл джерела інформації, де r – порядок резервного копіювання; $f_n^{'}(r-1)$ - резервна копія файлу $f_n(r-1)$; l – довжина сегменту. На першому ярусі відбувається операція паралельного розділення $f_n(r)$ на сегменти d_{kn}, при чому $l = const$ для будь-якого r. На другому

 ISSN 1562-9945

ярусі проводиться індексний пошук $d_{kn} \in f_n(r)$ та відповідного сегменту $d_{jn}^{'}$ у резервній копії (відбувається перебір по всіх k). На третьому ярусі відбувається по-бітне (або в залежності від реалізації, за хешем) порівняння відповідних сегментів. На четвертому ярусі, в разі, коли відповідний $d_n \in f_n(r) \cap d_n \notin f_n^{'}(r-1)$ або $d_{kn} \neq d_{jn}^{'}$ - відбувається процес копіювання (перенесення) сегменту даних у резервну копію та відповідна індексація. На виході маємо резервну копію $f_n^{'}(r)$, яка складається з M розподілених блоків.

Отже для алгоритму блочного резервного копіювання поле операцій буде таким $V = \{\upsilon_{11}...\upsilon_{1n}, \upsilon_{21}...\upsilon_{2m}, \upsilon_{31}...\upsilon_{3k}, \upsilon_{41}...\upsilon_{4l}\}$, де $n, m, k, l > 1$ кількість вузлів паралельної обробки (процесорів для однорідної паралельної системи, або комп'ютерів для розподіленої системи) на кожному ярусі.

$$G_p = (\{\upsilon_{11}...\upsilon_{1n}, \upsilon_{21}...\upsilon_{2m}, \upsilon_{31}...\upsilon_{3k}, \upsilon_{41}...\upsilon_{4l}\}, E_p) \quad (2)$$

Наданий граф є укрупненим – відображає основні механізми алгоритму та не включає в себе процедуру перебудови індексів, розрахунок хеш-сум для сегментів даних.

Для подальшого аналізу представимо алгоритм блочного резервного копіювання у вигляді діаграми розкладу. За початкові дані приймемо граф алгоритму. Для цього вісь ординат розіб'ємо на відрізки кожний з яких відповідає паралельно функціонуючому вузлу чи процесору.

У кожному інтервалі розміщено операції, що закріплені за вузлом чи процесором на даний момент часу. Зв'язки між операціями розміщено зліва та справа від операцій.

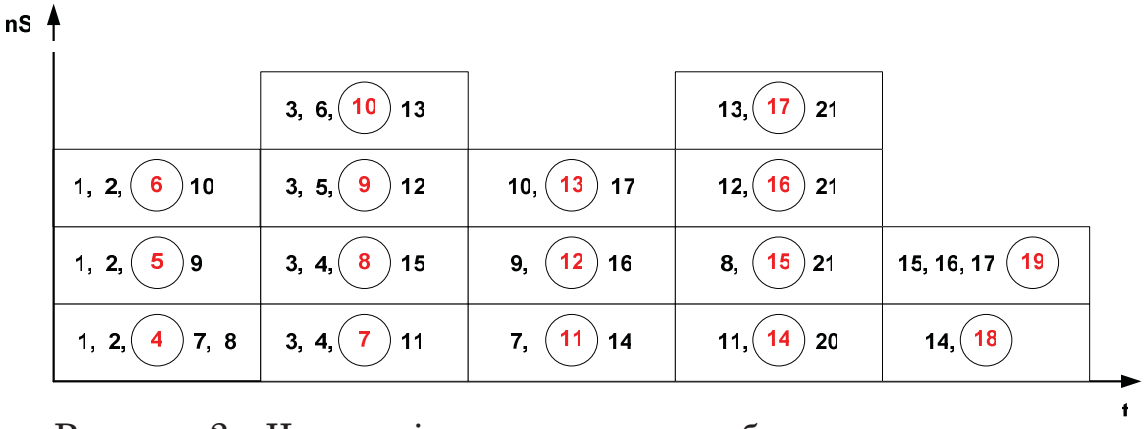

Рисунок 2 - Часова діаграма алгоритму блочного резервного копіювання

Наведений спосіб представлення є вичерпним наглядним представленням паралельного алгоритму для процедури резервного копіювання. Як видно на рис. 2. для проведення резервування використовується до 4-х процесорів/вузлів обробки.

Інформаційний граф G представимо матрицею прямування S (зведеної до нижньої трикутної форми див. рис. 3).

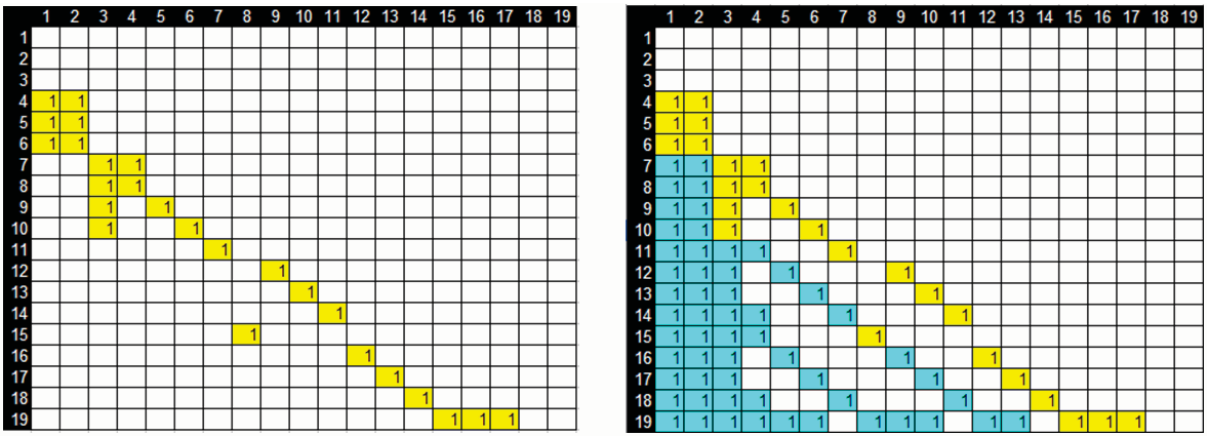

Рисунок 3 - Трикутна матриця прямування та доповнена матриця прямування з транзитивними зв'язками

Також на рис. 3. представлено матрицю прямування доповнену транзитивними зв'язками - $S^{'}$. Як видно з матриці $S^{'}$ граф не має петель.

Визначимо ранні та пізні строки виконання операторів. Для цього ярусно-паралельний граф представимо у вигляді інформаційного графа наступного вигляду:

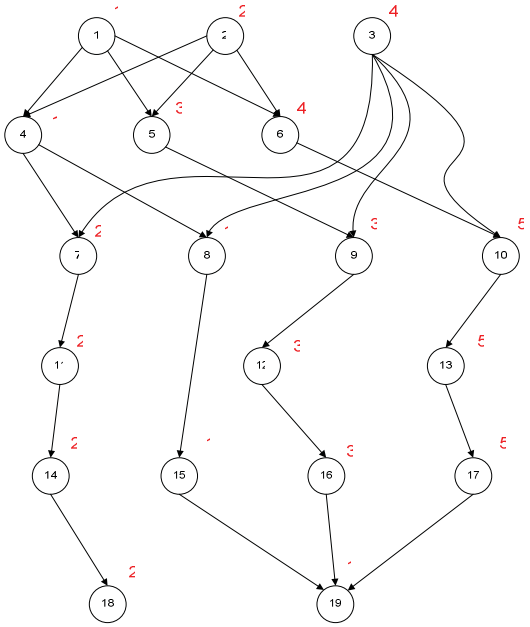

Рисунок 4 - Інформаційний граф

 ISSN 1562-9945

Кожній вершини інформаційного графа призначимо вагу. Інформаційний граф відображає порядок виконання операцій.

Знаходження ранніх строків виконання операторів T_{1i} будемо виконувати за наступним алгоритмом:

– Припускаємо, що першочергово $T_{11} = T_{12} = \ldots T_{1m} = 0$;

– Циклічно виконуємо огляд матриці S;

– Якщо строчка j не містить одиничних елементів, то $T_{1j} = t_j$;

– Якщо строчка j містить одиничні елементи, які формують множину $\{T_{1j(x)}\}, x = 1, \ldots, k_j$, то $T_{1j} = \max\{T_{1x}\} + t_j$

Критичний шлях знайдемо за наступною формулою:

$$T_{kr} = \max\{T_{11} \ldots . T_{1m}\} \quad (3)$$

Отже для нашого інформаційного графа (рис. 4) та матриці S проведемо розрахунки:

$T_{11} = t_1 = 1; T_{12} = t_2 = 2; T_{13} = t_3 = 4;$

$T_{14} = \max\{T_{11}, T_{12}\} + t_4 = 3; T_{15} = \max\{T_{11}, T_{12}\} + t_5 = 5; T_{16} = \max\{T_{11}, T_{12}\} + t_6 = 6;$

$T_{17} = \max\{T_{13}, T_{14}\} + t_7 = 6; T_{18} = \max\{T_{13}, T_{14}\} + t_8 = 5; T_{19} = \max\{T_{13}, T_{15}\} + t_9 = 8;$

$T_{1,10} = \max\{T_{13}, T_{16}\} + t_{10} = 11; T_{1,11} = T_{17} + t_{11} = 8; T_{1,12} = T_{19} + t_{12} = 11;$

$T_{1,13} = T_{1,10} + t_{13} = 16; T_{1,14} = T_{1,11} + t_{14} = 10; T_{1,15} = T_{1,8} + t_{15} = 6;$

$T_{1,16} = T_{1,12} + t_{16} = 14; T_{1,17} = T_{1,13} + t_{17} = 21; T_{1,18} = T_{1,14} + t_{18} = 12;$

$T_{1,19} = \max\{T_{1,15}, T_{1,16}, T_{1,17}\} + t_{19} = 22;$

$T_{kr} = \max\{T_{1,1}, \ldots., T_{1,19}\} = 22$

Пізні строки виконання операторів будемо шукати за таким алгоритмом:

– Припускаємо, що першочергово $\tau_{21}(T) = \tau_{22}(T) = \ldots \tau_{2m}(T) = 0$;

– Аналіз матриці виконуємо по стовпцям справа наліво. Якщо стовпець j не містить одиничних значень, то $\tau_{2j}(T) = T$;

– Якщо стовпець j містить одиничні елементи, які формують множину $\{\tau_{21}(T), \ldots, \tau_{2m}(T)\}$, та $\{\tau_{2j(x)}(T)\} \subset \{\tau_{21}(T), \ldots, \tau_{2m}(T)\}$, де $x = 1, \ldots, k_j$ то $\tau_{2j}(T) = \min\limits_{x=1,\ldots,k} \{\tau_{2j(x)}(T) - t_{j(x)}\}$

Допустимо, що $T = 23$, тоді розрахуємо $\{\tau_{21}(T), \ldots, \tau_{2m}(T)\}$:

$\tau_{2,19} = T = 23; \tau_{2,18} = T = 23; \tau_{2,17} = \tau_{2,19} - t_{19} = 22; \tau_{2,16} = \tau_{2,19} - t_{19} = 22;$

$\tau_{2,15} = \tau_{2,19} - t_{19} = 22; \tau_{2,14} = \tau_{2,18} - t_{18} = 21; \tau_{2,13} = \tau_{2,17} - t_{17} = 17;$

$\tau_{2,12} = \tau_{2,16} - t_{16} = 19; \tau_{2,11} = \tau_{2,14} - t_{14} = 19; \tau_{2,10} = \tau_{2,13} - t_{13} = 12;$

$\tau_{2,9} = \tau_{2,12} - t_{12} = 16; \tau_{2,8} = \tau_{2,15} - t_{15} = 21; \tau_{2,7} = \tau_{2,11} - t_{11} = 17;$

$\tau_{2,6} = \tau_{2,10} - t_{10} = 7; \tau_{2,5} = \tau_{2,9} - t_9 = 13; \tau_{2,4} = \min\{(\tau_{2,7} - t_7), (\tau_{2,8} - t_8)\} = \min\{15, 20\} = 15;$

$\tau_{2,3} = \min\{(\tau_{2,7} - t_7), (\tau_{2,8} - t_8), (\tau_{2,9} - t_9), (\tau_{2,10} - t_{10})\} = \min\{15, 20, 13, 7\} = 7;$

$\tau_{2,2} = \min\{(\tau_{2,4} - t_4), (\tau_{2,5} - t_5), (\tau_{2,6} - t_6)\} = \min\{14, 10, 3\} = 3;$

$\tau_{2,1} = \min\{(\tau_{2,4} - t_4), (\tau_{2,5} - t_5), (\tau_{2,6} - t_6)\} = \min\{14, 10, 3\} = 3;$

Побудуємо діаграму виконання робіт для ранніх та пізніх строків виконання операторів.

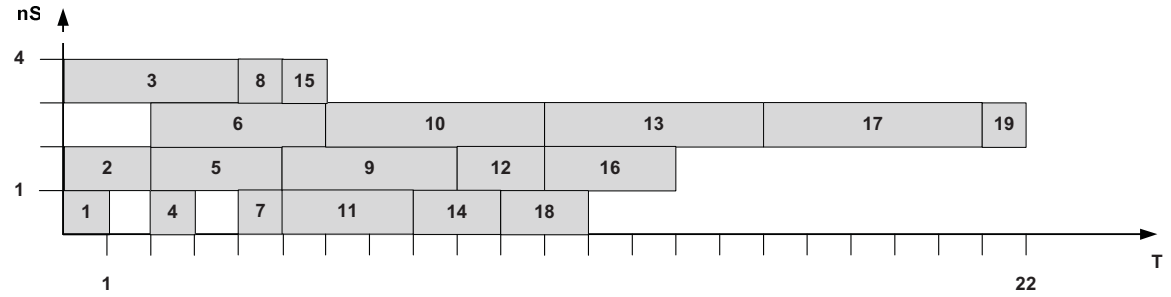

Рисунок - 5 Діаграма виконання робіт для ранніх строків виконання операторів

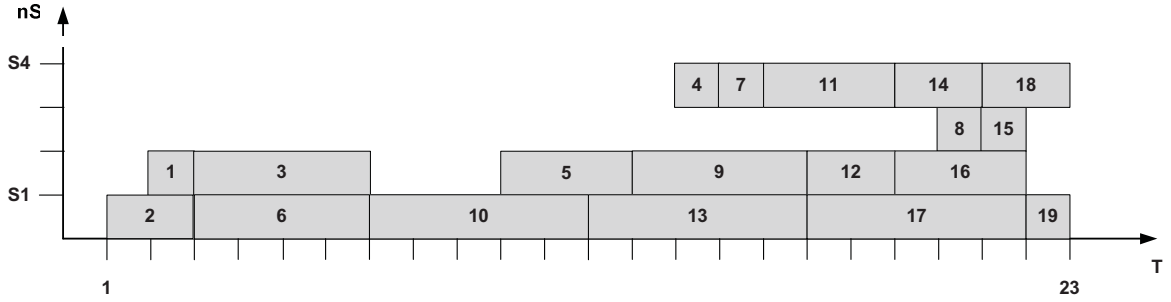

Рисунок 6 - Діаграма виконання робіт для пізніх строків виконання операторів

Означимо, що щільність завантаження при виконанні операцій буде визначатися функцією [5], [6]

$$F(\tau_1, \tau_2, \ldots \tau_m, t) = \sum_{j=1}^{m} f(\tau_j, t) \tag{4}$$

$$f(\tau_j, t) = \begin{cases} 1, t \in \lfloor \tau_j - t_j, \tau_j \rfloor \\ 0, t \notin \lfloor \tau_j - t_j, \tau_j \rfloor \end{cases} \tag{5}$$

де, τ_j – довільне значення моменту закінчення виконання j – го оператора.

Розрахуємо щільність завантаження для алгоритму паралельного блочного резервного копіювання представленого діаграмою виконання робіт для ранніх строків виконання операторів.

$F(1,2,4,1,3,4,2,1,3,5,2,3,5,2,1,3,5,2,1,t)$ при $t \in \lfloor 1,21 \rfloor$

$t=1, F=2; t=7, F=3; t=13, F=2; t=19, F=1$

$t=2, F=4; t=8, F=3; t=14, F=1; t=20, F=1$

$t=3, F=3; t=9, F=3; t=15, F=1; t=21, F=1$

$t=4, F=4; t=10, F=3; t=16, F=1$

$t=5, F=4; t=11, F=3; t=17, F=1$

$t=6, F=3; t=12, F=2; t=18, F=1$

Для алгоритму блочного резервного копіювання розрахуємо мінімальну кількість процесорів. Для цього розрахуємо максимальне значення щільності завантаження за формулами наведеними в [7], [8]

$$R = \max\{r_1, ..r_k\} \tag{6}$$

де, $r_i, i = \overline{1,k}$ – кількість робіт, що формують i – повну множину взаємно незалежних робіт, тоді

$$R = \max_{\tau_1,..\tau_m} F(\tau_1, ..\tau_m, t) \tag{7}$$

мінімальне число процесорів n однакової спеціалізації і продуктивності, здатних виконати даний алгоритм за $T \geq T_{kr}$ не перевищує $R = \max\{r_1, ..r_k\}$. Отже для множини $\{4,5,6\}$ $n=4$, для множини $\{7,8,9,10\}$ $n=3$, для множини $\{14,15,16,17\}$ $n=1$.

Висновок. Отримані дані дають змогу характеризувати паралельний алгоритм за ознаками мінімальної кількості процесорів необхідних для виконання резервного копіювання, часу виконання задачі, щільності завантаженості паралельної системи, що в свою чергу дозволяє побудувати ефективну систему резервного копіювання з використанням методів паралельних та хмарних обчислень. Обчислення свідчать про те, що введення паралельних алгоритмів до систем програмного резервування інформації має сенс при виконанні декількох умов: забезпечення достатньої кількості вузлів обробки (процесорів) та порівняно великого ступеню розпаралелення алгоритму (частина паралельних обчислень до загального обсягу обчислень).

ЛІТЕРАТУРА

1. R. Kolstad. A Next Step in Backup and Restore Technology. //In USENIX Proceedings of the 5th Conference on Large Installation Systems Administration, 1991, p. 73–78.
2. E. K. Lee and C. A. Thekkath. Petal. Distributed Virtual Disks. //In Seventh International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS-VII), 1996.
3. Шемет А.О. Застосування методів програмного резервування інформації: аспект резервного копіювання. Метод паралельного резервного копіювання. //Вісник Хмельницького національного університету. Технічні науки. 3.2011, - с. 221-225.
4. Шемет А.О. Паралелізм методів програмного резервування інформації. Метод паралельного блочного резервного копіювання. //Вісник академії митної служби України. Технічні науки, 1.2011, - с. 103-110.
5. J. da Silva, O. Gudmundsson, and D. Mosse. Performance of a Parallel Network Backup Manager. //In USENIX Conference Proceedings, 1992p. 17– 26.
6. Introduction to Algorithms. 2-nd edition / T. Cormen, C. Leiserson, R. Rivest, C. Stein. – London: The MIT Press, 2001. – 1180 p.
7. Воеводин В.В., Воеводин Вл. В. Параллельные вычисления. СПб.: БХВ-Петербург, 2002. – 608 с.
8. Немнюгин С., Стесик О. Параллельное программирование для многопроцессорных вычислительных систем. СПб.: БХВ-Петербург, 2002, - 400с.

 ISSN 1562-9945