УДК 004.021

Informatics and Mathematical Methods in Simulation
Vol. 8 (2018), No. 2, pp. 138-145

О БАЛАНСИРОВКЕ ВЫЧИСЛИТЕЛЬНОЙ НАГРУЗКИ ПРИ РАСПАРАЛЛЕЛИВАНИИ РЕШЕНИЯ ЗАДАЧИ НАХОЖДЕНИЯ ПОКРЫТИЯ

О.Н. Паулин

Одесский национальный политехнический университет,
просп. Шевченко, 1, Одесса, 65044, Украина; e-mail: paolenic@yandex.ua

Рассматривается известная комбинаторная задача нахождения покрытия методом теорем о свойствах таблицы покрытия. В более ранней работе автора приводится последовательное решение данной задачи, имеющей циклический характер. В новой работе автора предлагается способ распараллеливания решения этой задачи, основанный на свойстве независимости ветвей вычислительного процесса (подпроцессов); таким свойством обладают внешние циклы подпроцесов поиска ядерных/антиядерных строк и поглощающих столбцов/поглощаемых строк. Строится последовательно-параллельный информационный граф такого решения, приводится его описание. Особую важность имеет определение такого распараллеливания вычислительного процесса, при котором вычислительная нагрузка на процессоры является равномерной, то есть сбалансированной. На практике во многих случаях циклов имеет место постепенное снижение объёма вычислений в теле цикла от максимального до единичного, в результате чего нагрузка на процессоры становится существенно неравномерной. Рассматриваемая задача нахождения покрытия является таким случаем. В работе предлагается геометрическое представление вычислительной нагрузки. Описанный выше случай неравномерной нагрузки представляется треугольником вычислительной нагрузки. Показывается способ преобразования треугольника нагрузки в равновеликий прямоугольник, что обеспечивает эффективную балансировку нагрузки на процессоры в параллельной системе. Предлагается оценка недогруженности процессоров.

Ключевые слова: покрытие, таблица покрытия, метод теорем, параллельная система, параллельное решение задачи покрытия, вычислительный процесс, информационный граф, внешние циклы, балансировка вычислительной нагрузки, треугольник нагрузки, прямоугольник нагрузи, геометрическое преобразование фигур, оценка недогруженности процессоров

Введение

Практически важной в классе комбинаторных задач [2] является задача о покрытии, которую можно рассматривать как вариант задачи поиска в некотором конечном множестве определённых подмножеств с заданными свойствами. При этом находятся такие компоненты из множества возможных, которые покрывают заданную их совокупность оптимальным образом. Такая задача возникает, например, при необходимости выбора поставщиков при сборке сложного изделия. Ещё примеры: логические задачи о покрытии множеств, о поиске минимальных разбиений и др., при синтезе и анализе цифровых схем на программируемых логических матрицах [3].

Для описания задачи о покрытии используется таблица покрытия (ТП), представляющая собой матрицу отношений принадлежности элементов подмножеств A_i, $i = 1...m$, множества A элементам опорного множества B, $B = \{b_1,..., b_n\}$ таких, что $A_i \subseteq B$, $\bigcup_{i=1}^{m} A_i = B$.

Ниже рассматривается случай поиска кратчайшего покрытия, когда цены всех подмножеств A_i одинаковы и равны 1 [1].

Для решения практически важных задач о покрытии большой размерности необходима достаточно производительная параллельная система (ПС). В настоящее время имеется довольно большой спектр возможностей для выбора ПС различной архитектуры, начиная от многоядерных компьютеров и кончая кластерами [4, 5]. В данной работе этот аспект не является определяющим и потому он не рассматривается.

Различают два вида параллелизма: объектов/данных и независимых ветвей.

Параллелизм объектов или данных имеет место тогда, когда по одному и тому же алгоритму должна обрабатываться некоторая совокупность данных, поступающих в систему одновременно. Это может быть, например, обработка сигналов от радиолокационной станции или обработка информации от датчиков сигнализации в системе противопожарной безопасности, измеряющих одновременно один и тот же параметр. Это могут быть и чисто математические задачи, например задачи векторной алгебры – операции над векторами и матрицами, которые характеризуются некоторой совокупностью чисел. Решение задачи при этом в основном сводится к выполнению одинаковых операций над парами чисел. Очевидно, все эти операции могут выполняться одновременно и независимо друг от друга несколькими процессорами.

Параллелизм независимых ветвей – один из наиболее распространенных типов параллелизма в обработке информации. Суть его заключается в том, что при решении задачи могут быть выделены отдельные независимые части, которые при наличии нескольких процессоров могут выполняться параллельно [4, 5].

Отметим важный аспект распараллеливания решения задачи – балансировка нагрузки на процессоры, что, в конечном счёте, также ускоряет процесс решения задачи. Этому вопросу в литературе не уделено достаточного внимания.

Цель работы – ускорение процесса решения применительно к задаче нахождения покрытия за счёт распараллеливания её решения и балансировки нагрузки на процессоры.

Для достижения этой цели в работе решаются следующие *задачи*:

1. Распараллеливание решения задачи за счёт выделения в решении независимых участков (ветвей) вычислительного процесса (ВП);
2. Разработка наглядного представления вычислительной нагрузки на процессоры;
3. Разработка способа преобразования исходного представления к виду, при котором нагрузка на процессоры была бы как можно ближе к равномерной.

Основная часть

Ниже на примере решения задачи нахождения покрытия с использованием теорем о свойствах ТП [1] рассматривается способ перевода последовательной реализации решения в параллельную реализацию, основанную на свойстве независимости подпроцессов (ветвей) ВП. Такой подход соответствует модели параллельного программирования SPMD (Single Program – Multiple Data, одна программа – множество данных). При этом не ставилась цель оптимального распараллеливания – автор ограничился очевидным способом выделения участков ВП, которые могут быть выполнены одновременно, что свелось к распараллеливанию внешних циклов подпроцессов [4].

В то же время автор сосредоточился на поиске способа статической балансировки нагрузки на процессоры параллельной системы.

Приведём общие соображения об архитектуре параллельной системы. Анализ ВП показал, что он имеет следующие особенности: отсутствие межпроцессорных обменов

данными и интенсивная работа с ТП в параллельной части ВП; отсюда следует необходимость в локальной памяти для каждого процессора. Кроме того, в последовательной части необходимо выделение главного процессора для выполнения дополнительной работы по модификации ТП и координации работы процессоров ПС.

Само распараллеливания возможно при соблюдении условий независимости подпроцессов ВП. Независимыми ветвями (подпроцессами) являются такие части задачи, при выполнении которых реализуются следующие условия:

– ни одна из входных для подпроцесса величин не является выходной величиной другого подпроцесса (отсутствие функциональных связей);

– для любых двух подпроцессов не должна производиться запись в одни и те же ячейки памяти (недопустимость использования одних и тех же полей оперативной памяти);

– возможность выполнения одного подпроцесса не зависит от результатов или признаков, полученных при выполнении другого подпроцесса (независимость по управлению).

Рассмотрим общую идею решения задачи о предварительном покрытии с использованием теорем о свойствах таблицы покрытия (ТП).

На рис. 1 приведена схема алгоритма (СА) [1] решения данной задачи. Видно, что она состоит из блока ввода ТП, предопределённых подпроцессов нахождения ядерных строк (ЯС), антиядерных строк (АЯС), поглощающих столбцов (ПС) и поглощаемых строк (ПС’), блока формирования признака изменений в ТП, а также блока вывода результатов.

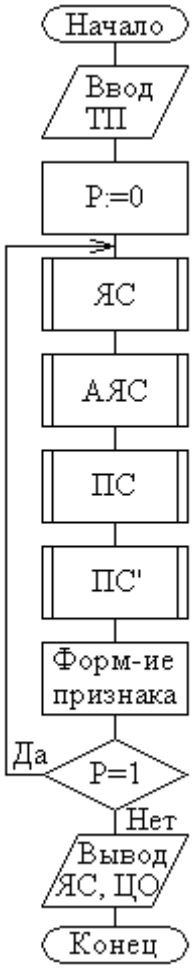


Рис. 1. СА поиска покрытия

Каждый из подпроцессов включают в себя 2 вложенных цикла, внешний и внутренний, перебора элементов ТП.

Процедура поиска решения задачи о покрытии является циклической: итерации (этапы ВП) прохода по предопределённым подпроцессам нахождения ЯС, АЯС, ПС, ПС’ повторяются, пока P=1, где P – признак изменения (сокращения) ТП за счёт вычеркивания строк/столбцов в соответствии с теоремами о свойствах ТП [1]. Если данная теорема выполнилась, то это приводит к сокращению ТП, при этом частный признак $p_i, i = 1...4,$ изменений в ТП принимает значение 1.

В блоке формирования общего признака P изменений в ТП вычисляется логическая формула

$$P = p_1 \vee p_2 \vee p_3 \vee p_4. \qquad (1)$$

Другими словами, если хотя бы один признак p_i принял значение 1, то процесс обхода подпроцессов повторяется; иначе выводится результат обработки ТП в виде списка ЯС и циклического остатка ТП.

Анализ ВП решения задачи покрытия [1] показывает, что он может быть представлен последовательно-параллельным информационным графом (рис. 2), причём распараллеливание достаточно просто осуществляется для подпроцессов, реализующих теоремы о нахождении ядерных строк (ЯС), антиядерных строк (АЯС), поглощающих столбцов (ПС) и поглощаемых строк (ПС’). Здесь обработка ТП осуществляется по самому внешнему циклу; количество процессоров в подпроцессах соответствует числу строк/столбцов текущей ТП.

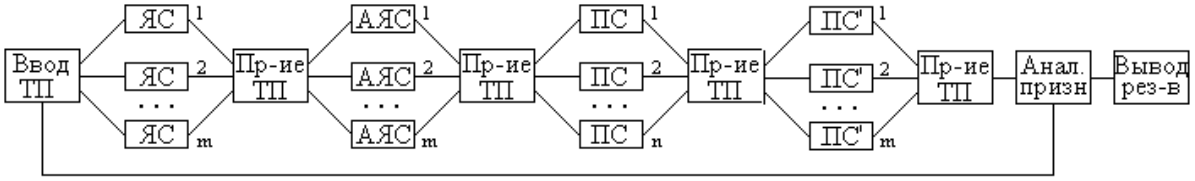


Рис. 2. Последовательно-параллельная организация вычислительного процесса

Последовательная часть графа представлена следующими блоками: ввод ТП, преобразование ТП, анализ признаков, вывод результатов.

В блоке *ввода* осуществляется рассылка всем узлам параллельной системы исходной ТП, имеющей m строк и n столбцов, а также всех текущих (модифицированных) ТП с новыми значениями m и n.

В блоках *преобразования* проводится слияние решений от отдельных процессоров, для чего собирается информация о возможных частных изменениях в ТП для каждого процессора – подпроцессы ЯС, АЯС, ПС, ПС’ сообщают блоку преобразований о необходимости вычёркивания строки/столбца и запоминания имени ЯС; формируется признак наличия изменений в ТП. Полученная таблица рассылается всем узлам параллельной системы на следующем этапе ВП.

В блоке *анализа* проводится вычисление по формуле (1) значения общего признака процесса P. Если P=1 (изменения в ТП произошли), то процесс запускается сначала для текущей ТП; в случае P=0 управление передаётся блоку вывода, в котором на экран монитора выводится список ядерных строк и циклический остаток ТП.

Рассмотрим распараллеливаемые фрагменты (подпроцессы) ВП [1]. Отметим прежде всего, что в каждом подпроцессе перед началом работы происходит обнуление признака p_i. Отметим далее, что узлы параллельной системы для всех подпроцессов могут выполнять свою часть работы в теле внешнего цикла независимо друг от друга.

– Подпроцесс *ЯС*: каждая строка анализируется на предмет наличия в ней особой единицы, т.е. единицы, которая в данном столбце является единственной; если строка

имеет особую единицу, то она является ядерной. При этом имя ЯС запоминается, а сама ЯС и покрываемые ею столбцы подлежат вычёркиванию; признак наличия изменений в ТП p_1 принимает значение 1. Эта информация передаётся в блок преобразования ТП.

– Подпроцесс *АЯС*: строка с нулевыми элементами является антиядерной; она подлежит удалению без запоминания. При этом признак наличия изменений в ТП p_2 принимает значение 1. Эта информация передаётся в блок преобразования ТП.

– Подпроцесс *ПС*: здесь проводится генерация пар (V, V') двоичных векторов и их сравнение на предмет поглощения; возможны 4 варианта: $V > V', V < V', V = V'$, V и V' несравнимы. Формируется сообщение о необходимости удаления поглощающего столбца; признак p_3 принимает значение 1. Эта информация передаётся в блок преобразования ТП.

– Подпроцесс *ПС'*: выполняется аналогично предыдущему, с тем отличием, что сравниваются строки-вектора. Формируется сообщение о необходимости удаления поглощаемой строки, если ищется кратчайшее покрытие; признак p_3 принимает значение 1. Эта информация передаётся в блок преобразования ТП.

Балансировка нагрузки. Более подробное рассмотрение подпроцесса нахождения поглощающего столбца/поглощаемой строки показывает, что при переходе к следующей итерации число сравниваемых пар векторов уменьшается на 1, что может быть представлено геометрическим образом (рис. 3). Назовём его треугольником нагрузки.

Если каждую строку (несколько строк, но одинаковое для всех процессоров число) обрабатывает отдельный процессор, то, очевидно, они находятся в неравных условиях: первый процессор должен проделать $n-1$ сравнение, а последний – лишь одно: обработать пару $(n-1, n)$. Для устранения такой неравномерности нагрузки нами предлагается организовать ВП иначе, представив нагрузку прямоугольником, равновеликим треугольнику, поскольку объём вычислений пропорционален площади фигуры, представляющей нагрузку.

Один из способов преобразования «треугольник – прямоугольник» представлен на рис. 4. В свою очередь, каждый из четырёх прямоугольников может быть разбит на одинаковое число k частей, так что в ВП могут участвовать $4k$ процессоров.

Важно, чтобы при похожих преобразованиях выдерживался принцип равенства общей площади прямоугольников площади исходного треугольника.

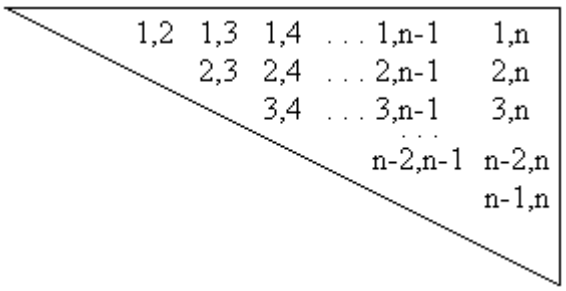


Рис. 3. Геометрическое представление процесса генерации пар векторов

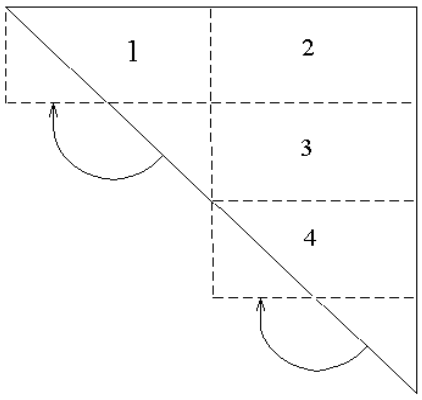
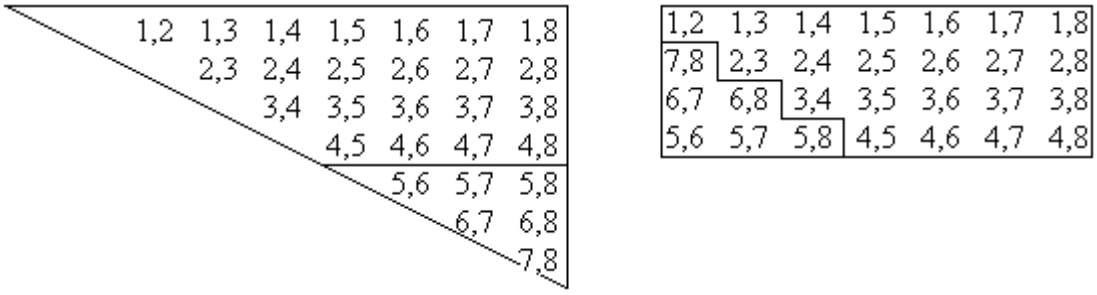


Рис. 4. Преобразование треугольника в прямоугольники

Рассмотрим пример преобразования треугольника нагрузки в прямоугольник нагрузки (рис. 5). Здесь выделенный горизонтальной линией малый треугольник пристраивается к трапеции слева, начиная с 5-й строки исходного массива пар (в общем

случае – с номером $(n+1/2)$, однако в обратном порядке. На рис. 5(б) показано данное перестроение (ступенчатая линия). Получившийся прямоугольник является образом равномерной нагрузки.



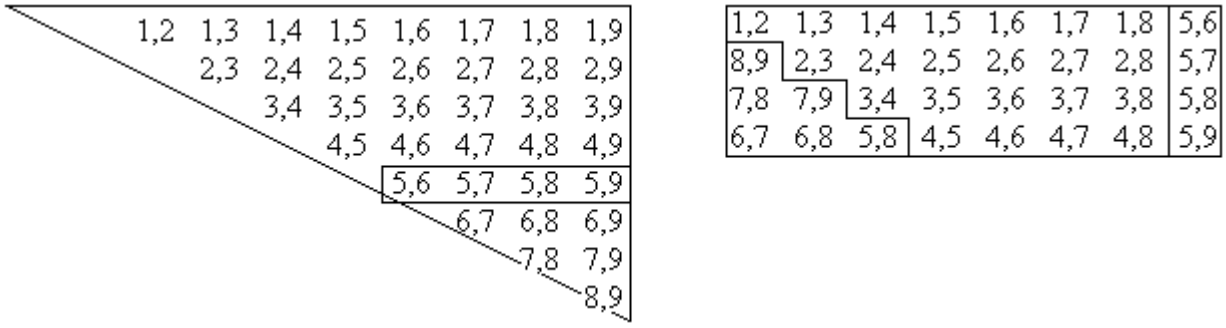
а б

Рис. 5. Преобразование для чётного n: а – треугольник нагрузки; б – прямоугольник нагрузки

Как видно из рис. 5, число обрабатываемых пар равно 28, необходимое число процессоров равно 4, на каждый процессор приходится по 7 пар.

Обобщим этот пример на случай произвольного чётного числа n. Имеем: число процессоров, необходимое для построчной обработки, равно $p = n/2$; на каждый процессор приходится по $k = (n-1)$ паре; всего количество обрабатываемых пар равно $N = n(n-1)/2$.

Анализ показал, что преобразование треугольника нагрузки в прямоугольник имеет особенность при нечётном n. Рассмотрим её на примере n = 9 (рис. 6).



а б

Рис. 6. Преобразование для нечётного n: а – треугольник нагрузки; б – прямоугольник нагрузки

В данном случае имеется особая строка с номером 5, которая выделена прямоугольником на рис. 6(а); она не вписывается в процедуру преобразования. Однако она может быть пристроена к прямоугольнику нагрузки справа в виде нового столбца – см. рис. 6(б). Остальные преобразования выполняются так же, как и в предыдущем случае. Параметры полученного прямоугольника таковы: число обрабатываемых пар равно 32, необходимое число процессоров равно 4, на каждый процессор приходится по 8 пар.

Обобщим этот пример на случай произвольного нечётного числа n. Имеем: число процессоров, необходимое для построчной обработки, равно $p = (n-1)/2$; на каждый

процессор приходится по $k = (n-1)$ паре; всего количество обрабатываемых пар равно $(n-1)^2/2$. Особая строка имеет номер $(n+1)/2$.

Обобщения для двух этих случаев отличаются формулами для определения числа p процессоров и общего количества обрабатываемых пар; однако эти формулы могут быть приведены к одному виду: $p = n\text{div}2$; $N = kp = k(n-1)$.

Отметим, что в реальности имеет место проблема недогруженности отдельных процессоров, например, из-за того, что число процессоров больше числа обрабатываемых объектов. Проиллюстрируем примером проблему некратности числа обрабатываемых объектов числу имеющихся процессоров. Пусть ТП содержит 82 строки и обрабатывается 12-ю параллельно работающими процессорами. Тогда вычислительный процесс будет протекать в шесть последовательных стадий $(82\text{div}12 = 6)$, причём на 7-й стадии 2 процессора будут простаивать $(12 - 82\text{mod}12 = 2)$.

Оценим разбаланс нагрузки следующей формулой $(N_и - N_р)/N_и$, где $N_и$ – объём работы в идеальном случае (без простоев процессоров), т.е. 84 единицы; $N_р$ – объём работы в реальном случае, т.е. 82 единицы. При этом конкретное время работы процессора принято равным 1. Для рассмотренного примера имеем 2,4 %. Этот результат можно считать вполне приемлемым.

Нагрузка сбалансирована, если $N_р = N_и$, т.е. в случае кратности объёма работы числу процессоров; некратность приводит к недогруженности вычислительной системы.

Выводы

Для параллельного решения задачи о покрытии использован известный способ [4] распараллеливания внешнего цикла для подпроцессов нахождения ядерных/антиядерных строк, а также поглощающих столбцов/поглощаемых строк. Разработано геометрическое представление нагрузки на процессоры параллельной системы в виде треугольника для неравномерной нагрузки и прямоугольника – для равномерной нагрузи. Предложен способ преобразования треугольника нагрузки в равновеликий прямоугольник нагрузки. Предложена оценка недогруженности системы для случая некратного соотношения количества ветвей ВП и процессоров.

Список литературы

1. Паулин, О.Н. Методы и алгоритмы покрытия (часть 2) / О.Н. Паулин // Інформатика та математичні методи в моделюванні. — Одеса, ОНПУ, 2017. — Том 4. — С. 333-338.
2. Рейнгольд, Э. Комбинаторные алгоритмы. Теория и практика / Э. Рейнгольд, Ю. Нивергельт, Н. Део. — М.: Мир, 1980. — 480 с.
3. Закревский, А.Д. Логический синтез каскадных схем / А.Д. Закревский. — М.: Наука, 1981. — 416 с.
4. Воеводин, В.В. Параллельные вычисления / В.В. Воеводин, Вл.В. Воеводин. — СПб: БХВ-Петербург, 2004. — 608 с.
5. Гергель, В.П. Основы параллельных вычислений для многопроцессорных вычислительных машин: Учебное пособие / В.П. Гергель, Р.Г. Стронгин. — Нижний Новгород: Изд-во ННГУ им. Н.И. Лобачевского, 2003. — 184 с.

ПРО БАЛАНСУВАННЯ ОБЧИСЛЮВАЛЬНОГО НАВАНТАЖЕННЯ ПРИ РОЗПАРАЛЕЛЮВАННІ РІШЕННЯ ЗАДАЧІ ЗНАХОДЖЕННЯ ПОКРИТТЯ

О.М. Паулін

Одеський національний політехнічний університет,
просп. Шевченка, 1, Одеса, 65044, Україна; e-mail: paolenic@yandex.ua

Розглядається відома комбінаторна задача знаходження покриття методом теорем про властивості таблиці покриття. У роботі автора «Методи та алгоритми покриття (частина 2)» [1] наводиться послідовне вирішення даної задачі, що має циклічний характер. У новій роботі автора пропонується спосіб розпаралелювання рішення цієї задачі, заснований на властивості незалежності гілок обчислювального процесу (підпроцесів); таку властивість мають зовнішні цикли підпроцесів пошуку ядерних/антиядерних рядків і поглинаючих стовпців або рядків, що поглинаються. Будується послідовно-паралельний інформаційний граф такого рішення, наводиться його опис. Особливу важливість має визначення такого розпаралелювання обчислювального процесу, при якому обчислювальне навантаження на процесори є рівномірним, тобто збалансованим. На практиці в багатьох випадках циклів має місце поступове зниження обсягу обчислень в тілі циклу від максимального до одиничного, в результаті чого навантаження на процесори стає суттєво нерівномірним. Розглянута задача знаходження покриття як раз і є таким випадком. В роботі запропоновано геометричне представлення обчислювального навантаження. Описаний вище випадок нерівномірного навантаження представляється трикутником обчислювального навантаження. Показується спосіб перетворення трикутника навантаження в рівновеликий прямокутник, що забезпечує ефективне балансування навантаження на процесори в паралельній системі. Пропонується оцінка недовантаження процесорів.

Ключові слова: покриття, таблиця покриття, метод теорем, паралельна система, паралельне рішення задачі покриття, обчислювальний процес, інформаційний граф, зовнішні цикли, балансування обчислювального навантаження, трикутник навантаження, прямокутник навантаження, геометричне перетворення фігур, оцінка недовантаження процесорів

ABOUT BALANCING THE COMPUTATIONAL LOAD WHEN THE PARALLELIZATION OF THE SOLUTION OF PROBLEM OF FINDING A COVERAGE

O.N. Paulin

Odesa National Polytechnic University,
1, Shevchenko Str., Odesa, 65044, Ukraine; e-mail: paolenic@yandex.ua

The well-known combinatorial problem of finding a coverage by the theorems method on the properties of the cover table is considered. The author's work "Methods and algorithms of coverage (part 2)" [1] provides a consistent solution to this problem, which has a cyclic character. In the new author's work the method of parallelization of the solution of this problem based on the property of independence of branches of computational process (subprocesses) is offered; external cycles of subprocesses possess such property search nuclear/antinuclear rows and columns of absorbing/absorption lines. A series-parallel information graph of such a solution is constructed and its description is given. Of particular importance is the definition of such parallelization of the computational process, in which the computational load on the processors is uniform, that is, balanced. In practice, in many cases of cycles, there is a gradual decrease in the volume of calculations in the body of the cycle from maximum to single, resulting in a load on the processors becomes significantly uneven. The considered problem of finding a covering is just such a case. The paper proposes a geometric representation of the computational load. The case of non-uniform load described above is represented by the triangle of the computational load. The method of converting the load triangle into an equal-sized rectangle is shown, which provides effective load balancing for processors in a parallel system. Assessment of underloaded processors is proposed.

Keywords: coverage, coverage table, the method of theorems, parallel system, parallel solution of problem of the covering, the computing process, information graph, the external cycle, balancing the computational load, triangle of load, rectangle of load, the geometric transformation of figures, assessment of underloaded processors