УДК 004.384:004.272:004.414.2

МЕТОД СЛУЧАЙНОГО ПЕРЕБОРА В ЗАДАЧЕ ПОСТРОЕНИЯ РАЗБИЕНИЙ ГРАФ-СХЕМ ПАРАЛЛЕЛЬНЫХ АЛГОРИТМОВ

Ватутин Э.И., Колясников Д.В., Мартынов И.А., Титов В.С.
Юго-Западный государственный университет, факультет информатики и вычислительной техники

Одним из перспективных направлений для построения систем логического управления (СЛУ) является их реализация в виде мультисистем, представленных коллективом однотипных логических контроллеров [1–4]. Подобные системы могут быть настроены на выполнение заданного параллельного алгоритма логического управления теоретически неограниченной сложности за счет его разбиения на множество последовательных блоков ограниченной сложности с их последующим размещением в модулях системы. Качество выбранного разбиения напрямую влияет на аппаратную сложность мультисистемы [5] и ее быстродействие, а сама задача отыскания (суб)оптимального разбиения имеет ярко выраженный комбинаторный характер и относится к классу *NP*. Отыскание оптимального решения на практике невозможно для граф-схем алгоритмов управления, содержащих $N > 10$ вершин, ввиду того, что временная асимптотика алгоритма полного перебора определяется числом Белла [6], стремительно растущим с ростом *N*, поэтому на практике для ее решения разработан и с успехом применяется ряд эвристических методов [7–12].

Данные методы могут быть разделены [1–4] на два больших класса: последовательные и итерационные. Для последовательных методов, к которым относятся метод С.И. Баранова [7–8] и его модификации [9–10], а также метод параллельно-последовательной декомпозиции [11–12], характерной особенностью является построение одного субоптимального решения, причем его качество определяется типом эвристики и подходом к синтезу разбиения. Так в методе С.И. Баранова [7–8], фактически реализующем жадный подход к построению решения, производится последовательное поблочное построение разбиения на основании весовой функции, учитывающей приращение числа логических условий, микроопераций и «веса» (числа команд в памяти контроллера) при добавлении очередной вершины в блок. При этом формирование очередного блока разбиения завершается при невозможности добавления вершин в текущий блок, в разбиение добавляется новый блок и производится его наполнение нераспределенными вершинами граф-схемы, процесс итерационно повторяется до тех пор, пока все вершины не будут распределены по блоками. Данный метод достаточно быстр, однако вычислительные эксперименты, выполненные в проекте добровольных распределенных вычислений Gerasim@home

[13] на платформе BOINC [14], показывают [15–17], что областью преимущественного использования метода является область слабых ограничений и сравнительно небольших граф-схемы алгоритмов управления (до 500–600 вершин). Модификация метода С.И. Баранова заключается в наложении ограничения на рассмотрение смежных вершин при формировании текущего блока разбиения [9–10]. Данная модификация не уступает прототипу по временным затратам на формирование решения, а предварительные результаты показывают, что данный подход показывает наилучшие результаты при ограничениях средней силы. Метод параллельно-последовательной декомпозиции [4, 11–12] реализует другую стратегию поиска разбиения: сперва производится построение разбиения граф-схемы на сечения [18–20], затем сечения, начиная с базового, разбиваются на субсечения и из них формируется искомое разбиение на базе построения и анализа таблиц включений [4, 21]. Данный метод в несколько раз проигрывает жадным подходам по времени построения решения, однако в наиболее сложных условиях при наличии сильных ограничений он способен давать решения наивысшего качества.

Итерационные методы, в отличие от последовательных, основаны на построении нескольких решений с последующим выбором наилучшего. Схематично обобщенный алгоритм их работы может быть представлен в следующем виде:

1. (инициализация) Положить качество решения $Q := \infty$, $i := 1$.
2. Сформировать *i*-е разбиение $\Gamma_i(G) = \{A_1, A_2, ..., A_H\}$ для заданной граф-схемы $G = \langle A, V \rangle$, где $A = \{a_1, a_2, ..., a_N\}$ – множество вершин граф-схемы алгоритма; $\forall A_j \in \Gamma_i(G)$: $A_j \subseteq A$, $j = \overline{1, H}$; $V = \{v_1, v_2, ..., v_M\}$ – множество дуг.
3. Оценить качество полученного разбиения $Q_i := J(\Gamma_i(G))$, где $J(.)$ – оценочная функция [22].
4. Если $Q_i < Q$, запомнить лучшее на данный момент разбиение $S := \Gamma_i(G)$, положить $Q := Q_i$.
5. Положить $i := i + 1$. Если $i \le C$, где C – число перебираемых вариантов решения, то перейти к п. 2.
6. Вернуть S в качестве результирующего разбиения.
7. Конец алгоритма.

Как видно из приведенного описания, отличие различных итерационных методов друг от друга заключается в способе получения очередного разбиения

Q_i. Временные затраты на поиск решения определяются как временем построения очередного решения (как правило, оно сверхлинейно зависит от размерности задачи N), так и числом перебираемых вариантов C (имеет место линейная зависимость). Тенденция изменения качества решений $Q(C)$ обычно имеет вид кривой, показанной на рис. 1.

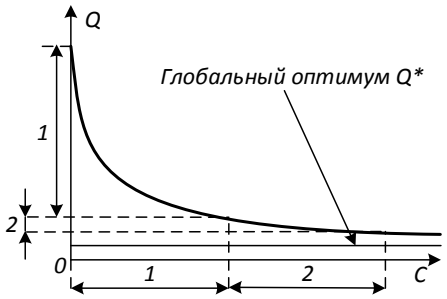

Рис. 1. Изменение качества решений Q итерационного метода с ростом числа итераций C

На начальном этапе перебора (отмечен на рис. 1 цифрой 1) обычно наблюдается стремительное улучшение качества решения, которое при $C \to \infty$ стремится к некоторому значению Q', обычно не достигающему глобального оптимума Q^*. На последующих этапах перебора (отмечены цифрой 2) увеличение числа итераций уже не приводит к значительному улучшению решения (обычно $\min\limits_{i=\overline{1,C}} Q_i$ отличается от Q' в лучшем случае на 1–2%), однако линейно увеличивает затраты вычислительного времени. Соответственно можно ввести понятие оптимального значения числа итераций C^*, при увеличении которого качество решений не претерпевает существенных изменений при использовании данного итерационного метода (как показывают вычислительные эксперименты, $C^* \simeq 10^3 \div 10^5$ для различных задач дискретной комбинаторной оптимизации, например [23]). Если временные затраты на поиск решения ограничены (например, требуется оперативно произвести реконфигурацию системы жесткого реального времени), то число итераций может быть снижено, однако при этом следует ожидать ухудшения качества получаемых решений.

Итерационные методы решения задачи поиска разбиения характеризуются асимптотической временной сложностью порядка $O\left(N^3\right)-O\left(N^5\right)$, а соответствующие им программные реализации требуют как минимум в 100 больших временных затрат по сравнению с последовательными подходами, что по видимому является причиной их слабой распространенности на практике [1–2]. Однако у методов данного класса есть и существенное преимущество, связанное с тем, что с точки зрения параллельного программирования описанная задача поиска решения является слабосвязной, т.к. она не требует обменов данными при построении различных решений $\Gamma_i\left(G\right)$ и $\Gamma_j\left(G\right)$, $i \neq j$ (что может быть неверным для некоторых методов улучшения решений, формально также являющихся итерационными, или генетических эволюционных алгоритмов). Данная особенность, в свою очередь, позволяет распараллеливание вычислительного процесса с использованием широкого спектра вычислительных средств: многоядерных процессоров и многопроцессорных машин, а также кластеров, суперкомпьютеров и грид-систем на их основе. Сдержанный оптимизм вызывает возможность использования для поиска решений видеокарт в рамках концепции GPGPU, однако комбинаторные оптимизационные задачи рассматриваемого класса не являются вычислительными и обычно требуют большого числа нерегулярных обращений в память, что может вызвать существенную деградацию реальной производительности видеокарты [24].

При построении разбиения может быть использована стратегия случайного перебора, алгоритм которой представлен ниже.

1. (инициализация) Положить разбиение равным пустому множеству: $\Gamma\left(G\right) \coloneqq \varnothing$, $H \coloneqq 0$; множество нерассмотренных вершин равным множеству вершин граф-схемы: $\tilde{A} \coloneqq A$.
2. Случайным образом выбрать из множества нерассмотренных вершин текущую вершину $a_i \in \tilde{A}$.
3. Сформировать подмножество блоков разбиения $\Gamma^+ \subseteq \Gamma$, в которые допустимо включение вершины a_i (включение не нарушает ограничений). Для каждого из блоков $A_j \in \Gamma^+$ сформировать оценку $q_j \coloneqq r\alpha$, где r – очередное псевдослучайное число в диапазоне $\left[0;1\right]$, получаемое с использованием соответствующего генератора, $0 \leq \alpha \leq 1$ – числовой коэффициент, характеризующий вероятность попадания вершины в существующий блок. Добавить к ним оценку $q_0 \coloneqq r\left(1-\alpha\right)$, соответствующую включению вершины в новый блок.

4. Выбрать максимальную оценку $j^* = \arg\max_{j=0,\overline{|\Gamma^+|}} q_j$. Если $j^* = 0$, добавить в разбиение новый блок $\Gamma := \Gamma \cup A_{H+1}$, $A_{H+1} = \{a_i\}$, образованный вершиной a_i, и положить $H := H + 1$, в противном случае произвести включение вершины a_i в блок A_{j^*}: $A_{j^*} := A_{j^*} \cup \{a_i\}$.
5. Исключить вершину a_i из множества нерассмотренных вершин: $\tilde{A} := \tilde{A} \setminus \{a_i\}$. Если множество нерассмотренных вершин не пусто $\left(\left|\tilde{A}\right| \neq 0\right)$, перейти к п. 2.
6. Конец алгоритма.

Алгоритм реализует построение одного псевдослучайного разбиения. На практике, следуя описанной выше итерационной стратегии, с использованием приведенного алгоритма производится построение C разбиений, из которых впоследствии выбирается лучшее по критерию $J\left(\Gamma_i(G)\right) \to \min$. Алгоритм включает в себя настроечный параметр α, который определяет вероятность попадания вершин в новые блоки. При его большом значении разбиения характеризуются большим числом блоков H, которое, в свою очередь, ведет к увеличению других показателей качества.

Приведенный алгоритм построения одного разбиения является последовательным и не допускает распараллеливания, т.к. формирование решения производится путем последовательного рассмотрения вершин граф-схемы алгоритма и имеет место RAW-зависимость по формируемому разбиению Γ. Однако различные итерации алгоритма для различных выборок псевдослучайных чисел, получаемых путем использования различных начальных значений генератора псевдослучайных чисел, могут быть выполнены параллельно. Если в распоряжении имеется K вычислителей, то каждый из них производит построение группы из $\left\lceil \frac{C}{K} \right\rceil$ разбиений с выбором лучшего. Далее из найденных K решений выбирается наилучшее, что может быть реализовано по принципу двоичного дерева за время порядка $\lceil \log_2 K \rceil$.

Литература

1. Организация и синтез микропрограммных мультимикроконтроллеров / Зотов И.В. и др.; Курск: Изд-во «Курск», 1999. 368 с.
2. Архитектура параллельных логических мультиконтроллеров / Емельянов С.Г., Зотов И.В., Титов В.С. М: Высшая школа, 2009. 233 с.

3. Комбинаторно-логические задачи синтеза разбиений параллельных алгоритмов логического управления при проектировании логических мультиконтроллеров / Э.И. Ватутин, И.В. Зотов, В.С. Титов и др.; Курск: изд-во КурскГТУ, 2010. 200 с.
4. Ватутин Э.И. Проектирование логических мультиконтроллеров. Синтез разбиений параллельных граф-схем алгоритмов. Saarbrücken: Lambert Academic Publishing, 2011. 292 с.
5. Ватутин Э.И., Титов В.С. Структурно-параметрическая оптимизация систем логического управления с использованием добровольных распределенных вычислений // Известия Юго-Западного государственного университета. Серия «Управление, вычислительная техника, информатика. Медицинское приборостроение». 2012. № 2. Ч. 1. С. 12–17.
6. https://ru.wikipedia.org/wiki/Числа_Белла
7. Баранов С.И., Журавина Л.Н., Песчанский В.А. Обобщенный метод декомпозиции граф-схем алгоритмов // А и ВТ. 1982. № 5. С. 43–51.
8. Ватутин Э.И. Библиотека функций построения разбиений методом С.И. Баранова с жадным последовательным формированием блоков // Свидетельство о государственной регистрации программы для ЭВМ № 2010612902 от 28.04.10.
9. Ватутин Э.И., Леонов М.Е. Использование смежной окрестности при жадном последовательном формировании блоков разбиения граф-схем параллельных алгоритмов // Известия высших учебных заведений. Приборостроение. 2013. Т. 56. № 6. С. 30–35.
10. Ватутин Э.И., Титов В.С. Библиотека функций для построения разбиений с использованием смежной жадной стратегии и последовательным формированием блоков // Свидетельство о государственной регистрации программы для ЭВМ № 2013619395 от 03.10.13.
11. Ватутин Э.И., Зотов И.В. Метод формирования субоптимальных разбиений параллельных управляющих алгоритмов // Параллельные вычисления и задачи управления (PACO’04). М.: ИПУ РАН, 2004. С. 884–917.
12. Ватутин Э.И., Зотов И.В. Параллельно-последовательный метод формирования субоптимальных разбиений параллельных управляющих алгоритмов // Свидетельство об официальной регистрации программы для ЭВМ № 2005613091 от 28.11.05.
13. http://gerasim.boinc.ru
14. http://boinc.berkeley.edu
15. Ватутин Э.И., Титов В.С. Сравнение методов синтеза разбиений параллельных алгоритмов логического управления с использованием двухпараметрических диаграмм // Распознавание – 2012. Курск: изд-

во ЮЗГУ, 2012. С. 138–140.
16. Ватутин Э.И., Титов В.С. Сравнение методов синтеза разбиений граф-схем параллельных алгоритмов с использованием двумерных диаграмм // Известия Юго-Западного государственного университета. Курск, 2012. № 3 (42), 2012. С. 66–74.
17. Ватутин Э.И., Титов В.С. Использование добровольных распределенных вычислений на платформе BOINC для анализа качества разбиений граф-схем параллельных алгоритмов // Параллельные вычисления и задачи управления (PACO'12). М.: ИПУ РАН, 2012. Т. 2. С. 37–54.
18. Поиск базового сечения в задаче разбиения параллельных алгоритмов / Ватутин Э.И., Зотов И.В.; КГТУ. Курск, 2003. 30 с. Рук. деп. в ВИНИТИ 24.11.03 № 2036-В2003.
19. Ватутин Э.И., Зотов И.В., Титов В.С. Построение множества сечений в задаче оптимального разбиения параллельных управляющих алгоритмов // Известия ТулГУ. Вычислительная техника. Информационные технологии. Системы управления. Тула, 2003. Т. 1. В. 2. С. 70–77.
20. Ватутин Э.И., Зотов И.В., Титов В.С. Выявление изоморфных вхождений *R*-выражений при построении множества сечений параллельных алгоритмов логического управления // Информационно-измерительные и управляющие системы. № 11, Т. 7. М.: «Радиотехника», 2009. С. 49–56.
21. Ватутин Э.И., Зотов И.В. Построение блоков разбиения в задаче декомпозиции параллельных управляющих алгоритмов / Материалы и упрочняющие технологии – 2003, Курск. гос. техн. ун-т. Курск, 2003. – Т. 2. С. 38–42.
22. Ватутин Э.И. Оценка качества разбиений параллельных управляющих алгоритмов на последовательные подалгоритмы с использованием весовой функции // Интеллектуальные и информационные системы (Интеллект-2005). Тула, 2005. С. 29–30.
23. Ватутин Э.И., Романченко А.С., Титов В.С. Исследование влияния порядка рассмотрения пар на качество расписаний при использовании жадного подхода // Известия Юго-Западного государственного университета. 2013. № 1 (46). С. 58–64.
24. Параллельные вычисления на GPU. Архитектура и программная модель CUDA / Боресков А.В., Харламов А.А. Марковский Н.Д. и др. М.: изд-во Московского университета, 2012. 336 с.