Компьютерные инструменты в образовании, 2018
№ 2: 31–38
УДК: 004.21
http://ipo.spb.ru/journal

ОСОБЕННОСТИ РАСПРЕДЕЛЁННЫХ ВЫЧИСЛЕНИЙ, УЧИТЫВАЕМЫЕ В МЕТОДАХ ОПТИМИЗАЦИИ АЛГОРИТМОВ ПО ОБЪЕМУ МЕЖПРОЦЕССОРНЫХ ПЕРЕДАЧ

Аль-Марди М. Х. [1]

[1] Санкт-Петербургский государственный электротехнический университет «ЛЭТИ» им. В.И. Ульянова (Ленина), Санкт-Петербург, Россия

Аннотация

В связи с развитием новых технологий и расширением сферы применения вычислительной техники и сетей, в частности, с появлением Интернета вещей все чаще возникает потребность в применении совокупности вычислительных устройств для решения одной задачи. При этом чаще всего эти устройства являются географически распределенными и отличаются друг от друга техническими характеристиками. В данной статье рассматривается возможность адаптации методов параллельных вычислений, изначально предназначенных для систем с раздельной памятью, к распределенным системам. Рассматриваются особенности распределенных систем, способы представления таких систем и алгоритмов на этих системах. В статье предлагается подход к представлению алгоритмов с учетом особенностей распределенной системы в виде проекции информационного графа алгоритма на граф взаимосвязей узлов. Этот подход позволяет исследовать алгоритм в статическом режиме, не перебирая при этом все состояния сети и алгоритма.

Ключевые слова: *оптимизация, алгоритм, информационный граф, мощность узла, пропускная способность, время выполнения, операция, процесс, единица времени, время передачи.*

Цитирование: Аль-Марди М. Х. Особенности распределённых вычислений, учитываемые в методах оптимизации алгоритмов по объему межпроцессорных передач // Компьютерные инструменты в образовании. 2018. № 2. С. 31–38.

1. ВВЕДЕНИЕ

Имеется большое количество важнейших задач, решение которых требует использования огромных вычислительных мощностей, зачастую недоступных для современных вычислительных систем. Одним из показателей качества параллельной программы является плотность загрузки вычислительных узлов. Временные задержки при передаче данных по каналам связи от одного процессора к другому приводят к суммарно длительным простоям процессоров и увеличению в целом времени работы алгоритма [1, 2].

Цель построения и эквивалентного преобразования информационного графа заключается в том, чтобы минимизировать общее время выполнения программы за счет наилучшего распределения задач процессорам и организации последовательности выполнения задач на каждом процессоре [3, 7]. Само построение расписания решения задачи

выполняется таким образом, чтобы сохранить приоритетность среди всех задач, выполняемых в распределенной среде [4]. При этом последовательность выполнения подзадач на каждом процессоре называется расписанием программы, а общее время выполнения параллельной программы обычно называется длиной расписания или makespan [5, 6].

В данной статье предлагаются несколько решений задачи получения расписания выполнения алгоритма, эффективного по ряду параметров, таких как мощность узла, пропускная способность канала, время передачи данных и др. в вычислительной распределенной среде с применением методов оптимизации, разработанных для компьютеров с MPP архитектурой [8].

2. ОСНОВНЫЕ ПАРАМЕТРЫ, УЧИТЫВАЕМЫЕ В МЕТОДАХ ОПТИМИЗАЦИИ АЛГОРИТМОВ ПО ОБЪЕМУ МЕЖПРОЦЕССОРНЫХ ПЕРЕДАЧ

Основной отличительной чертой разработанных для параллельных вычислений методов было условие, что все вычислительные узлы работают с одинаковой мощностью и поэтому время срабатывания одной и той же операции на разных узлах предполагалось одинаковым. В распределенных вычислениях часто это далеко не так, поэтому для алгоритмов в распределенных вычислениях добавляется сразу ряд дополнительных параметров. Эти параметры могут быть разными в зависимости от архитектуры сети. Например, они могут включать:

- *Мощность узла.* Методики, по которой рассчитывается мощность, могут быть самыми разными. Их суть не влияет на оптимизацию алгоритмов. Важно только, чтобы для всех узлов мощность была рассчитана по одной методике и следовательно ранжирование узлов по этому параметру было корректным.
- *Пропускная способность канала.* Компьютеры обмениваются информацией с использованием каналов связи различной физической природы: кабельных, оптоволоконных, радиоканалов и др. Под пропускной способностью канала понимается максимальная скорость передачи информации по каналу связи в единицу времени. Под скоростью передачи информации понимается скорость информационного потока, то есть количество информации, передаваемое за единицу времени.
- *Время передачи.* Оно будет зависеть от физических параметров сети и длины канала. Неважно сколько и какие это будут параметры. Суть проблемы в том, что их будет больше одного и они будут числовыми.

Для построения расписания выполнения алгоритма по заданному информационному графу необходимо построить 2 графа:

1. Граф взаимосвязей распределённой системы вычислений. Этот граф является неориентированным, взвешенным. Причем, он будет не просто взвешенным, а совокупностью весов $(P_1, P_2, \ldots, P_m)$ (где $P_i = p_{ij}, i = 1..m, j = 1..n$, m — число весов, n — число вершин графа), каждый из которых будет влиять на расписание выполнения алгоритма. Пример такого графа приведен на рис. 1.

2. Информационный граф алгоритма в параллельной форме, по которой можно однозначно воспроизвести расписание выполнения алгоритма.

Вся сложность этого подхода заключается в том, что для построения второго графа (информационного графа алгоритма в параллельной форме) необходимо произвести многократную проекцию начального информационного графа алгоритма на граф взаимосвязей.

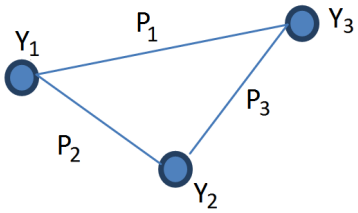

Рис. 1. Пример взвешенного графа системы распределенных вычислений, состоящей из трех узлов (Y_i — узлы, $P_i = \{p_i, l_i, d_i\}$, p_i — производительность узла, l_i — пропускная способность, d_i — длина пути)

Пример. Пусть дан граф, представленный на рис. 2 а. После применения к этому графу алгоритма разделения вершин по ярусам, получим следующие группы вершин по ярусам $M_1(1,2,3), M_2(5,4,6), M_3(8,7), M_4(9)$. В соответствии с этими группами будет построен информационный граф в параллельной форме в первом приближении, то есть без учета времени выполнения операций и особенностей распределенной системы. Этот граф приведен на рис. 2 б.

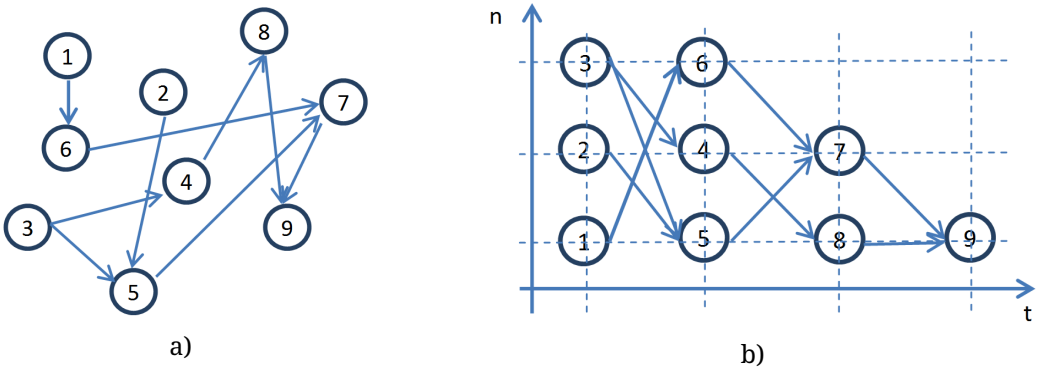

Рис. 2. Пример информационного графа алгоритма: a) начальный граф, b) приведенный к параллельной форме граф

Пусть операции, соответствующие вершинам, выполняются согласно времени $T = t_i = 3,2,4,5,2,3,4,3,3$, где i — номер вершины, $i = 1..9$. Для операций также известен максимальный объем данных, получаемый на выходе из операции: $V = v_i = 13,4,6,1,3,6,9,5,8$, где i — номер вершины, $i = 1..9$.

По узлам доступна следующая информация:

— производительность j-го узла $P = p_j = 7,5,3$;
— длина пути между j-м и $(j+1)$-м узлами $D = d_j = 5,7,15$;
— пропускная способность сети между j-м и $(j+1)$-м узлами $L = l_j = 10,8,9$.

Здесь j — номер узла, $j = 1..3$

Тогда в начальный момент времени возможна следующая проекция (рис. 3) информационного графа (рис. 2 b) на граф взаимосвязей узлов (рис. 1). Проведем несложные расчеты. Пусть на j-м ярусе n_j вершин, граф взаимосвязей состоит из n_y узлов. На каждом ярусе возможно $n_j * n_y$ всевозможных комбинаций из узлов и операций. Тогда общее количество проекций информационного графа на граф взаимосвязей будет составлять:

$$G_p = \prod_{j=1}^{m} n_j n_y ,$$

где m — число ярусов.

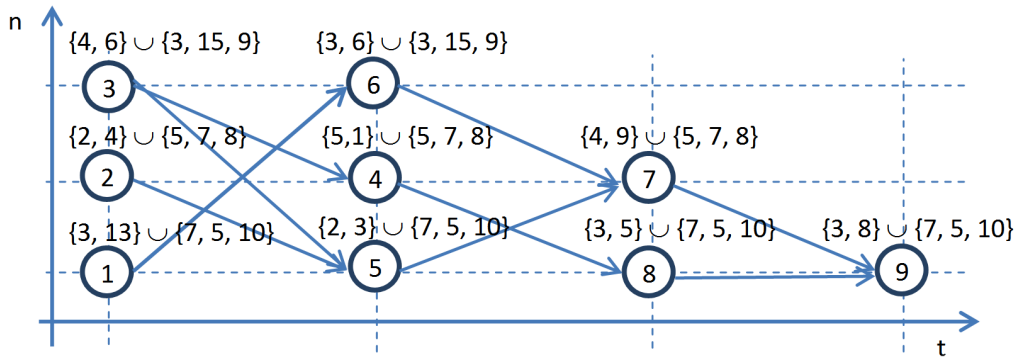

Рис. 3. Пример проекции информационного графа алгоритма на граф взаимосвязей узлов

Для информационного графа (рис. 2 b) и графа взаимосвязей (рис. 1) число проекций составит:

$$G_p = (3 \cdot 3) \cdot (3 \cdot 3) \cdot (2 \cdot 3) \cdot (1 \cdot 3) = 1458.$$

Для такого маленького графа, как на рис. 2 число проекций является очень большим. Проанализировать все эти проекции на предмет наиболее эффективной по времени выполнения и числу задействованных ресурсов простым перебором вручную невозможно. Очевидно, что с ростом числа вершин информационного графа и числа различных по мощности, расположению и другим параметрам узлов число проекций будет увеличиваться с большой скоростью. И для информационного графа с числом вершин более 100 и вычислительной системой более чем с 10 узлами число проекций уйдет за 1 млн. Анализ такого большого числа проекций перебором даже с помощью вычислительной техники займет очень много времени.

3. РЕШЕНИЕ ПРОБЛЕМЫ

Решений этой проблемы есть несколько, например:

1. Сократить число проекций путем сокращения для каждой вершины списка возможных узлов. Например, результатом операции, соответствующей вершине 7, является совокупность данных объемом $v_7 = 9$. Если эта операция выполняется на 1 узле, то на следующем ярусе из адресатов данных можно исключить узел под номером 2, для которого пропускная способность участка сети между 1-м и 2-м узлами равна всего 8.

Недостатками этого подхода являются:

- Трудоемкость расчетов числа проекций, которые остаются в поиске оптимального расписания.
- Усложнение метода поиска оптимального расписания за счет добавления G_p условий на проверку, подлежит ли узел рассмотрению или нет.
- Незначительное уменьшение числа проекций. Например, для графов 2 и 1. G_p уменьшится не более чем на 10.

- Непрактичность. Число добавляемых условий проверки востребованности узла для каждой операции будет намного выше, чем число проекций, которые убираются из рассмотрения. Трудоемкость, таким образом, возрастет.

2. Предварительная обработка информационного графа без учета времени выполнения операций, но с учетом межпроцессорных передач. Это позволит сократить число передач и убрать из рассмотрения все проекции, на которых данные не передаются от узла к узлу.

В качестве предварительной обработки информационного графа подойдёт, например, разработанный нами метод оптимизации алгоритма по объему межпроцессорных передач. Метод опубликован в предыдущей статье [9] и заключается в следующем:

1. Положить за основу группы вершин, соответствующих ярусам, полученные произвольным способом. Лучше, если эти группы будут получены формализованным способом, например, методом оптимизации информационного графа по ширине с помощью матрицы или списка смежности.
2. Процесс перестановки вершин начинается с последней группы. Допустим всего групп m, тогда номер очередной группы $k = m$. В первую очередь необходимо в расписание поставить (с учётом групп) вершины, у которых бинарная связь. И только потом расставлять вершины с множественными связями, так как в этом случае существование пузыря неизбежно.
3. В k-й группе выбрать первую вершину. Считать номер позиции этой вершины в группе равным 1: $i = 1$.
4. Сравнить вершину M_{k_i} (где i — номер позиции вершины в группе) с вершинами предыдущей $(k-1)$-й группы. Если в $(k-1)$-й группе существует вершина (M_{k-1_j}, где j — номер позиции вершины в группе, $j >= i$), напрямую связанная в информационном графе ребром с данной вершиной, то вершину M_{k-1_j} необходимо переместить в своей группе в i-ю позицию. Если в $(k-1)$-й группе нет вершины, связанной с вершиной M_{k_i}, то перейти к шагу 6.
5. Если $k > 2$, то $k = k - 1$ и перейти к шагу 4.
6. Если в m-й группе перебраны еще не все вершины, то $k = m$, $i = i + 1$ и перейти к шагу 4.
7. Если в последней группе перебраны все вершины и $m > 2$, то $m = m - 1$ и перейти к шагу 6.
8. Если $m = 1$, то **конец метода**.

После применения этого метода оптимизации по объему межпроцессорных передач к информационному графу (рис. 2 b) получим граф (рис. 4).

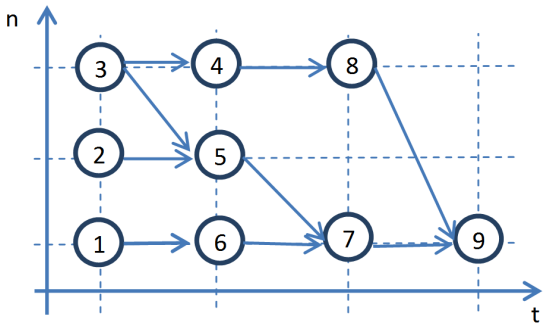

Рис. 4. Информационный граф, оптимизированный по числу межпроцессорных передач

В передаче данных в результате не нуждаются вершины 1, 2, 4, 6, 7, 9.
В таблице 1 приведены по каждой вершине значения комбинаций.

Таблица 1. Число комбинаций для вершин

Вершины	1	2	3	4	5	6	7	8	9
Комбинации	3	3	3	1	1	1	1	1	1

Операции 5 и 6 зависят от операций, которые в передаче данных не нуждаются, так как обладают простыми связями («точка-точка»), то есть операции 5 и 6 зависят каждая только от результата одной операции. Поэтому выбора для операций 5 и 6 нет. Они, чтобы избежать передачи данных и потери времени на передачу, должны выполняться на тех же узлах, что и операции 1 и 2. Аналогичная ситуация складывается для операций 7, 8 и 9.

Для операции 4 просто не остается выбора, так как из трех доступных узлов два уже заняты.

Соответственно, число комбинаций вершин, а следовательно, и число проекций будет равно:

$$G_p = 3 \cdot 3 = 9.$$

Таким образом, для выбора оптимального расписания необходимо оценить всего 9 графов. Это в 162 раза меньше общего числа проекций.

Недостатки этого решения следующие:

- Не для каждого информационного графа может быть достигнуто такое значительное сокращение числа проекций.
- Часто при передаче информации на более высокопроизводительный узел время выполнения операции может быть меньше даже с учетом затрат на процесс передачи данных по сравнению со временем выполнения на этом же узле, но маломощном. Поэтому принятие этого решения дает локальное оптимальное расписание, которое может не совпадать с глобальным.

3. Разработка метода оптимизации расписания поярусно. Подбор узлов на первых двух ярусах, а далее выбор узлов с учетом тех, что уже зафиксированы для вершин предшествующих ярусов. Этот метод будет более эффективным по трудоемкости по сравнению с методом перебора, но его недостатком является также отсутствие гарантии, что найденное расписание будет глобальным оптимумом.

4. ЗАКЛЮЧЕНИЕ

В данной статье рассмотрены особенности распределенной вычислительной системы и ее отличие от систем с MPP архитектурой. Приведено описание графа взаимосвязей вычислительной системы и информационного графа алгоритма. Показано, что сложность проектирования расписания алгоритма в распределенной среде заключается в том, что для построения второго графа (информационного графа алгоритма в параллельной форме) необходимо произвести многократную проекцию начального информационного графа алгоритма на граф взаимосвязей. В результате обработке будет подлежать очень большое количество графов, что сразу скажется на времени построения расписания. Одним из решений этой проблемы является применение метода оптимизации алгоритма по объему межпроцессорных передач. В статье показано,

что применение этого метода, позволяет в десятки раз ускорить процесс построения расписания в распределенной среде.

Для наглядности все методы проиллюстрированы на одном примере. Но следует отметить, что описанные в статье решения были все протестированы с помощью специальной программы на более чем 100 контрольных примерах.

Список литературы

1. *Abramov O. V., Katueva Ya.* Multivariant analysis and stochastic optimization using parallel processing techniques. // Management problems. 2003. № 4. P. 11–15.
2. *Jordan H. F., Alaghband F.* Fundamentals of Parallel Processing. Pearson Education, Inc., Upper Saddle River, NJ, 2003. P. 578.
3. *Drake D. E., Hougardy S.* A linear-time approximation algorithm for weighted matchings in graphs // ACM Transactions on Algorithms. 2005. № 1. P. 107–122.
4. *Hu Chen.* MPIPP: An Automatic Profileguided Parallel Process Placement Toolset for SMP Clusters and Multiclusters / Hu.Chen // Proceedings of the 20th annual international conference on Supercomputing. New York, NY, USA. 2006. P. 353–360.
5. *Rauber N., Runger G.* Parallel Programming: for Multicore and Cluster Systems. / N. Rauber, G. Runger. Chemnitz, Germany: Springer, 2010. 450 p.
6. *Gergel V. P.* Lectures of Parallel Programming: Proc. Benefit / Gergel V. P., Fursov V. A. Samara State Aerospace University Publishing House, 2009. 163 p.
7. *Voevodin V. V., Voevodin Vl. V.* Parallel computing. St. Petersburg: BHV-Petersburg, 2002. 608 p.
8. *Шичкина Ю. А.* Сокращение высоты информационного графа параллельных программ // Научно-технические ведомости СПбГПУ. 2009. № 3 (80). С. 148–152.
9. *Шичкина Ю. А., Аль-Марди М. Х.* Метод оптимизации параллельного алгоритма за счет уменьшения объема межпроцессорной передачи информации // Сборник известий СПб ГЭТУ. 2015. № 10. С. 15–23.

Поступила в редакцию 05.03.2018, окончательный вариант — 19.04.2018.

Computer tools in education, 2018
№ 2: 31–38
http://ipo.spb.ru/journal

FEATURES OF DISTRIBUTED COMPUTING, TAKEN INTO ACCOUNT IN OPTIMIZATION METHODS OF ALGORITHMS OPTIMIZATION FOR THE VOLUME OF INTERPROCESSOR TRANSMISSIONS

Al-Mardi M. H. A.[1]

[1]Saint-Petersburg Electrotechnical University, Saint Petersburg, Russia

Abstract

In connection with the development of new technologies and the expansion of the use of computers and networks, in particular with the advent of the Internet of things, there is an increasing need to use a combination of computing devices to solve one problem. In this

case, most often these devices are geographically distributed and differ from each other by technical characteristics. This article explores the possibility of adapting the methods of parallel computations originally intended for systems with separate memory to distributed systems. The features of distributed systems, ways of representing such systems and algorithms on these systems are considered. The article proposes an approach to the representation of algorithms taking into account the features of a distributed system in the form of a projection of an information graph of the algorithm onto a interconnection graph. This approach allows us to investigate the algorithm in static mode without going through all the network and algorithm states.

Keywords: *optimization, algorithm, information graph, node power, throughput capacity, execution time, operation, process, processor, unit of time, transmission time.*

Citation: M. H. A. Al-Mardi, "Features of distributed computing, taken into account in optimization methods of algorithms optimization for the volume of interprocessor transmissions," *Computer tools in education*, no. 2, pp. 31–38, 2018 (in Russian).

Received 05.03.2018, the final version — 19.04.2018.

Al-Mardi Mohammed Haidar Awadh, PhD student at department of Computer Science and Engineering–4, ETU «LETI»; 197376 , St. Petersburg, Russian Federation, ul. Professora Popova 5, building 2, Department of Computer Science and Engineering–4, almardi-md@mail.ru

Аль-Марди Мохаммед Хайдар Авадх, аспирант кафедры вычислительной техники–4 СПбГЭТУ «ЛЭТИ»; 197376, Санкт-Петербург, ул. Профессора Попова, д. 5, корп. 2, кафедра ВТ–4, almardi-md@mail.ru