Алгоритмы и графы в теории прямых разложений алгебраических структур, приложения к распараллеливанию вычислений

Е. А. Благовещенская[1,2]
Петербургский государственный университет путей сообщения Императора Александра I,
Санкт-Петербургский политехнический университет Петра Великого
kblag2002@yahoo.com

В. В. Гарбарук
Петербургский государственный университет путей сообщения Императора Александра I
vmkaf@pgups.ru

Н. Г. Павлова[1]
Математический институт им. С.М. Никольского, РУДН
natasharussia@mail.ru

С. А. Тихомиров
Ярославский государственный педагогический университет им. К.Д.Ушинского
satikhomirov@mail.ru

А. Е. Трифонов
Филиал акционерного общества «Концерн радиостроения «Вега» в г. Санкт-Петербурге
algisothal@gmail.com

В. В. Яковлев
Петербургский государственный университет путей сообщения Императора Александра I
yakovlev@pgups.ru

Аннотация. **Абелевы группы без кручения, являясь одной из первичных алгебраических структур, допускают алгоритмический подход к изучению их различных прямых разложений. Используемые графы имеют ярусно-параллельную форму и оказываются применимыми к задачам распараллеливания.**

Ключевые слова: теория групп; абелевы группы; параллельные вычисления; графы

I. Введение

Важной характеристикой производительности компьютера является алгоритмическая эффективность, так как это имеет решающее значение при обработке терабайт информации за малое время и при необходимости часто повторять этот процесс. Улучшение производительности начинается с разумного алгоритма, который описывает ядро выполняемой программы. А значит важно управлять доступными потоками процессора наиболее эффективным способом, известным программисту. Хотя существуют удобные расширения C++ для параллельных вычислений, существует более эффективный алгоритм, взятый из теории групп и представленный в этой статье.

Работа выполнена при финансовой [1]поддержке РФФИ, проект №17-01-00849, и при [2]поддержке госзадания Минобрнауки РФ, проект № 11.5861.2017/БЧ

II. Постановка задачи

Есть K процессоров, которые должны выполнить n вычислительных задач, состоящих из независимых цепочек, в каждой из которых вычисления производятся последовательно. Необходимо найти последовательно-параллельную форму алгоритма, позволяющего минимизировать временные затраты на выполнение расчетов. В настоящей работе исследователи рассматривают известные алгоритмы из теории прямых разложений абелевых групп без кручения, которые могут быть использованы для составления параллельного расписания. Также иллюстрируется алгоритм с помощью программы, реализованной на языке C++.

III. Существующий метод

Одним из наиболее распространенных существующих методов распараллеливания частей программы – это расширение процесса компиляции кода C++ под названием OpenMP. Он легок в освоении, не требует полностью переписывать программу для реализации параллелизма, в связи с чем весьма удобен. Несмотря на простоту использования, OpenMP не реализует наиболее эффективные методы параллельной разработки. По сути, OpenMP поддерживает спецификацию программы для реализации взаимодействия потоков. Если таких потоков несколько, то OpenMP будет разделять ресурсы, включая адресное пространство соответствующего процесса. Для каждого отдельного процесса требуются свои собственные ресурсы, то есть он должен иметь дескриптор в системе и место в памяти для хранения переменных, относящихся к программе (включая регистры). Несколько потоков могут использоваться в одном процессе. Такие процессы, используемые на нескольких процессорах, могут работать одновременно во время выполнения основной программы. Каждый раз, когда инструкция распараллеливания встречается в потоке, который выполняется программой, он создается по модели «fork-join» [1]. Это означает, что программа начинает выполнять свой первый назначенный поток так же, как она это делает с обычной программой. Каждый раз, когда инструкция распараллеливания встречается в потоке в OpenMP при его выполнении, она интерпретируется командами управления потоками. По щкщнчании исполнения лидирующий процесс продолжает свою работу, остальные завершаются.

IV. Теоретическая часть

A. *Графы*

В параллельной системе в любой момент времени может быть большое количество задач, которые должны быть выполнены одновременно и независимо друг от друга. В такого рода системах программа и ее входные данные определяются специальным образом: это содержание операций и последовательность этих операций. Таким образом, требуется предварительное условие, которое должно быть удовлетворено операциями и их последовательностью выполнения, поскольку необходимо гарантировать, что каждая часть программы получает необходимые аргументы, в связи с тем что они становятся доступными лишь после прохождения необходимых предыдущих шагов в программе. Это означает, что программист устанавливает частичный порядок, организующий получение результатов на разных процессорах.

Мы получили визуализацию алгоритма с помощью графов. Пусть существует непустое множество V, тогда множество $V^{(2)}$ – это множеством всех его двухэлементных подмножеств. Пара (V, E), где E – любое подмножество $V^{(2)}$, называется графом. Элементы из E называются ребрами графа. Ориентированный граф – это пара (V, A), где V – множество вершин, а A – множество ориентированных ребер, называемых дугами. Если $a=(v_1, v_2)$ – дуга, то вершины v_1, v_2 называются ее началом и концом соответственно. Мы будем обозначать дуги стрелками, которые показывают направление от начала к концу.

Введем понятие строго линейного графа. Пусть это будет такой ориентированный граф с n вершинами, индексированными от 1 до n и содержащими дуги от 1 до (n-1), которые связывают вершины i и i+1, где i=1,2,...,n-1. Почти линейный граф получается путем удаления некоторых дуг из линейного. Спиралевидный граф по модулю r – это граф, который получается из линейного путем удаления дуг (i, i+1) для i > r-1, в частности, при факторизации модуля r все дуги (0, 1) удаляются (предполагается, что r < n).

Спиралевидный граф по модулю r является частным случаем почти линейного графа. Последний можно интерпретировать как последовательное выполнение отдельных независимых цепей операций. В работе представлен алгоритм преобразования почти линейного графа в спиралевидный граф по модулю r, где r – длина самой длинной цепи (ее длина – количество операций внутри этой цепи). Получаемый граф является графом алгоритма, иллюстрирующим параллелизм выполняемых операций в предположении, что n не превышает произведение mr, где m – число процессоров.

На рис. 1 приведен пример почти линейного графа.

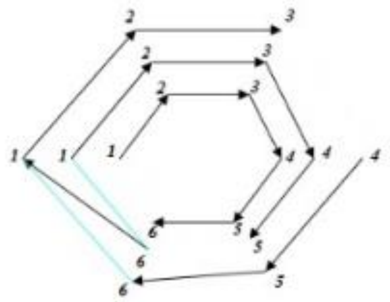

Рис. 1. Почти линейный граф

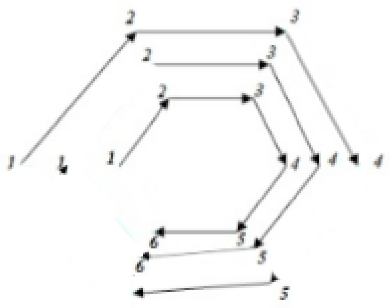

Рис. 2. Почти линейный граф по модулю 6

На рис. 2 параллельность цепочек проявляется в том, что операции выполняются независимо друг от друга. Такой график можно получить из фиксированных входящих данных. В следующем случае общее число задач равно 18, длины цепочек 6, 5, 4 и 3; цепочки выполняются параллельно. Каждая вершина представляет собой одну операцию.

Таким образом, граф алгоритма является его информационным ядром, формирующим последовательность независимых цепочек таким образом, что они выполняются одновременно в достаточно короткий промежуток времени. Идея рассмотреть эти конкретные типы графов заимствована из теории групп, в частности, теории почти вполне разложимых.

B. *Алгебра*

Традиционным методом алгебраических исследований является разложение аддитивных структур в прямую сумму неразложимых объектов. Мы рассматриваем класс групп без кручения конечного ранга, в частности блочножестких почти вполне разложимых с циклическим регуляторным фактором (так называемых crq-групп).

Это означает, что группа X содержит почти вполне разложимую характеристическую подгруппу A с попарно несравнимыми или полностью идентичными компонентами ранга 1, изоморфными подкольцам кольца Q, так называемого «регулятора», фактор-группа относительно которой X/A является конечной циклической группой [2].

Теорема. Пусть $n > r > 2$ – натуральные числа. Тогда существует блочно-жесткая crq-группа X ранга n, разложимая в прямую сумму неразложимых слагаемых рангов $r_1, r_2, \ldots, r_s$ для любых разбиений $n = r_1 + r_2 + \ldots + r_s$, в которых наибольшие слагаемые r_1 равняются числу r.

C. Алгоритм

В задаче с параллельными вычислениями за основу программы берется почти линейный граф. Затем множество чисел его вершин факторизуется по модулю r. Если граф представляет собой спиралевидный граф по модулю r, то задача распараллеливания цепи автоматически является решенной. В противном случае рассмотрим номера вершин, которые являются сравнимыми по модулю r и лежат на одних и тех же радиусах некоторых концентрических окружностей. Чем дальше окружность от центра, тем больше её порядковый номер. Подходящими вращениями окружностей относительно друг друга достигается ситуация, в которой точки с отсутствующими дугами располагаются в одном и том же месте (между вершинами 0 и 1). На рис. 3 показан граф алгоритма параллельного выполнения цепей операций, полученный при количестве процессоров, равном 4, и цепей следующих длин: 19, 15, 15, 15, 3, 2, 2, 2, 1, 1 (количество операций равно 75).

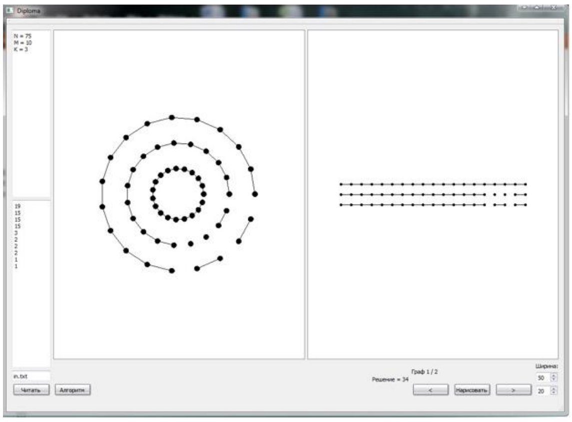

Рис. 3. Почти линейный граф по модулю 19

Список литературы

[1] Voevodin V.V., Voevodin V.L. V. Parallel computing. SPb: BHV-Petersburg, 2002. 608 S. ISBN 5-94157-160-7.

[2] Blagoveshchenskaya E., Kunetz D. Direct Decomposition Theory of Torsion-Free Abelian Groups of Finite Rank: Graph Method (2018) Lobachevskii Journal of Mathematics, 39 (1), pp. 29-34.