ВЗВЕШЕННЫЕ ГРАФЫ В ПАРАЛЛЕЛЬНЫХ ВЫЧИСЛЕНИЯХ

УДК 519.711.3, 681.51.015, 681.3.01,658.012.011.56:658.512

Ю.А.Шичкина кандидат технических наук (Братский государственный университет)

Аннотация

В настоящее время параллельное программирование является областью, в которой быстро появляются новые языки программирования, вычислительные системы, новые методы программирования. Тем не менее, проблема разбиения программы на процессы по-прежнему ложится на разработчика. В данной статье рассмотрены аспекты оптимизации параллельного алгоритма с помощью взвешенного информационного графа по двум параметрам: числу процессоров и времени выполнения. Основой алгоритма является перераспределение процессов по процессорам и укрупнение процессов с помощью их объединения.

Введение

Эффективность параллельного алгоритма зависит от многих параметров. Одним из таких параметров является равномерная загрузка процессоров вычислительной системы. При этом важно с одной стороны сохранив высоту параллельного алгоритма минимизировать его ширину. Другим важным параметром является время выполнения процессов. Если предполагать, что время выполнения всех процессов, отраженных в информационном графе в виде отдельных вершин, одинаково, тогда ориентированный ациклический информационный граф является наиболее подходящим инструментом для построения оптимального по высоте и ширине параллельного алгоритма. Но на практике такие условия – крайняя редкость. Следовательно, из-за неравномерности выполнения отдельных процессов, часть процессоров будет простаивать в ожидании результатов от своих предшественников. Поэтому, сама собой напрашивается замена ориентированного ациклического информационного графа его взвешенным аналогом, в котором в качестве весов будет выступать время выполнения процессов.

Временные диаграммы

Рассмотрим взвешенный информационный граф некоторого вычислительного процесса (рис.1).

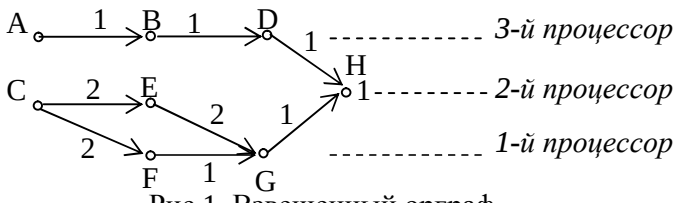

Рис.1. Взвешенный орграф.

При анализе вычислительного процесса, соответствующего данному информационному графу возникают следующие вопросы:

- Насколько непрерывно работают три процессора, задействованные в вычислительном процессе системе? Сколько времени простаивает каждый процессор в процессе работы алгоритма?
- Можно ли сократить общее время работы алгоритмы путем уменьшения времени простоев процессоров?
- Можно ли сократить ширину алгоритма путем уменьшения времени простоев процессоров за счет объединения мелких (по времени работы) операций в более крупные?

Для ответа на эти вопросы построим сначала временную диаграмму (рис.2), где t – время выполнения каждого процесса, n – номер процессора, задействованного в вычислении.

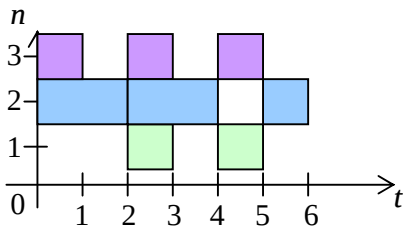

Рис.2. Временная диаграмма работы процессоров.

Построенная временная диаграмма показывает:

- Суммарное время работы алгоритма равно 6 ед.
- Первый процессор простаивает дважды и суммарное время простаивания составляет 3 ед., второй процессор простаивает единожды 1 ед., третий – дважды и суммарное время простаивания равно 2 ед.

Таким образом, оптимизация процесса только по числу процессоров не является конечным результатом улучшения качества параллельного алгоритма.

Из рис.2 видно, что два процесса, вычисляемы на одном процессоре (A и B) можно объединить в один, тем самым, произведя укрупнение схемы вычислений (рис.3).

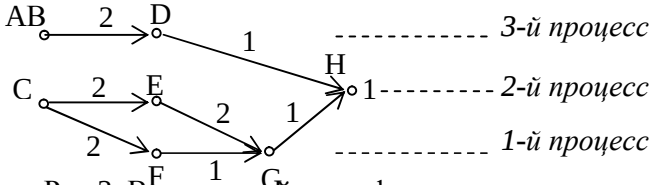

Рис.3. Взвешенный орграф.

Временная диаграмма при этом примет вид (рис.4):

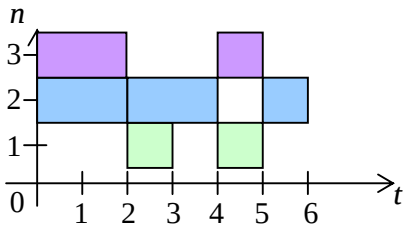

Рис.4. Временная диаграмма работы процессов.

Вследствие этого укрупнения, на первом ярусе два процессора работают одинаково. Алгоритм оптимизации полученного информационного графа (рис.3) с помощью матричного алгоритма по числу процессов позволяет преобразовать его к следующему виду (рис.5):

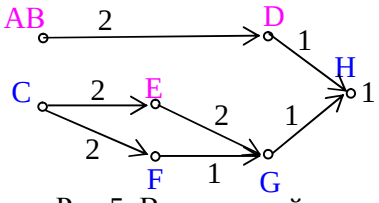

Рис.5. Взвешенный орграф.

Ширина алгоритма при этом становится равной 2. Следодательно, в вычислениях могут быть задействованы два процессора, что увеличит плотность вычисления и сделает применение вычислительной техники более эффективным. Временная диаграмма работы процессоров примет вид (рис.6).

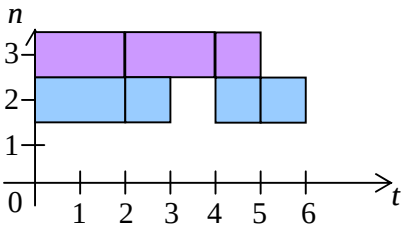

Рис.6. Временная диаграмма работы процессоров.

Вследствие этого, можно ответить на третий из поставленных вопросов: ширину информационного графа возможно сократить.

Автоматизировать процесс оптимизации информационного графа по швысоте с помощью временных диаграмм достаточно сложно. Формализовать этот процесс можно, если заменить информационный граф на *граф работы процессоров* (рис.7).

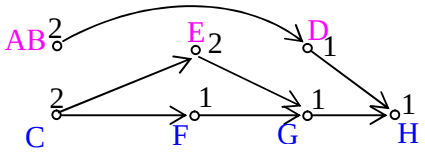

Рис.7. Граф работы процессоров.

При этом остается открытым вопрос об эффективности используемой памяти вычислительной системы. И скорее всего, схема использования памяти в данном случае будет хуже, чем в первоначальном варианте (рис.1).

Алгоритм оптимизации высоты информационного графа

Матрица смежности, соответствующая информационному графу с рис.1 имеет вид:

C=

	A	B	C	D	E	F	G	H
A	0	1	0	0	0	0	0	0
B	0	0	0	1	0	0	0	0
C	0	0	0	0	2	2	0	0
D	0	0	0	0	0	0	0	1
E	0	0	0	0	0	0	2	0
F	0	0	0	0	0	0	1	0
G	0	0	0	0	0	0	0	1
H	0	0	0	0	0	0	0	0

Т.к. каждая вершина графа соответствует одному процессу, выполняемому за определенное время, то в любой отдельно взятой строке все ненулевые элементы равны между собой. Это свойство взвешенного по времени информационного графа.

Алгоритм оптимизации высоты графа, является продолжением матричного алгоритма оптимизации ширины графа и заключается в следующем:

1) Разбить вершины графа на группы с помощью алгоритма оптимизации ширины информационного графа.

2) Начиная с первого яруса, найти группу вершин с разными временными значениями. Отметить ее.

3) В найденной группе найти максимальное значение по времени.

4) Отметить вершину, временное значение которой меньше максимального и рассмотреть возможность ее объединения с вершиной из следующей группы по принципу:

Если в строке матрицы смежности, соответствующей отмеченной вершине есть ненулевой элемент, соответствующий вершине из следующей группы и в столбце, соответствующем этой вершине, отсутствуют ненулевые элементы в строках отмеченной группы, то отмеченную вершину можно объединить с проанализированной вершиной из следующей группы.

Формально, это условие выглядит так:

Пусть v_i^1 *- отмеченная вершина группы* G_1, v_j^2 *- вершина из следующей группы* G_2, v_{ij} *- элементы матрицы смежности, стоящие на пересечении строк вершин группы* G_1 *и j-го столбца, тогда:*

$$v_i^1 = v_i^1 + v_j^2, \text{ если } \left(v_j^1 \neq 0\right) \cap \left(v_{ij} = 0\right)$$

5) Повторить четвертый шаг для всех вершин отмеченной группы.

6) Повторить шаги 3-5 для всех следующих групп, кроме последней.

7) Повторять алгоритм до тех пор пока не останется ни одного объединения.

Пример. Рассмотрим процесс работы алгоритма на информационном графе с рис. 1.

1) Разобьем вершины графа на группы с помощью матричного алгоритма оптимизации ширины информационного графа. В результате можно выделить следующие группы: AC, BEF, DG, H.

Ширина графа равна трем, высота – четырем, общее число вершин – восемь. Следовательно, оптимальная ширина при данной высоте равна двум (8/4=2). Данная ширина не может быть достигнута, т.к. выходная вершина всего одна, и, следовательно, на оставшиеся три яруса будет приходиться семь вершин и реальная высота: *trunc(7/3)=trunc(2.3)=3.* Поэтому, вторую часть матричного алгоритма оптимизации ширины графа можно не применять, т.к. реальная и оптимальная ширина.

2) Возьмем первую группу. Время выполнения операций A и C разное: 1 и 2.

3) В отмеченной группе найдем максимальное значение по времени: $t_C = 2$.

4) Отметим вершину, временное значение которой меньше максимального : A. Рассмотрим возможность ее объединения с вершиной из следующей группы. В следующей группе четыре вершины. В строке, соответствующей вершине A существует только одна единица – в столбце вершины B. В этом столбце других ненулевых элементов нет. Следовательно, вершину B можно переместить в отмеченную группу и объединить с вершиной A.

5) Больше подобных вершин нет. Алгоритм закончен.

Процесс выполнения шагов 2-5 отображен в таблице 1.

Таблица 1.

Алгоритм оптимизации высоты информационного графа

Группы	Время	Укрупнение 1		Укрупнение 2
A,C	1,2	(AB),C	2,2	
B,E,F	1,2,1	E,F	2,1	невозможно
D,G	1,1	D,G	1,1	
H	1	H	1	

В результате выполнения данного алгоритма получен граф с рис.7.

Ширина графа стала равной 2, что увеличивает эффективность работы вычислительной системы.

Заключение

Апробация полученных алгоритмов показала, что применение алгоритма оптимизации информационного графа по ширине, как правило, сокращая ширину графа, оставляет достаточно много простоев процессоров. Добиться наилучшего результата можно, только проведя после первого шага оптимизацию по времени или применив комбинированный метод. В зависимости от структуры информационного графа в разных случаях превалируют или тот или другой метод. В результате применения алгоритма оптимизации информационного графа по времени выполнения в ряде случаев могут сокращаться как высота алгоритма, так и ширина.

Список литературы

1. Лупин С.А., Посыпкин М.А. Технологии параллельного программирования. – М.: ИД «Форум»: ИНФРА-М, 2008. – 208с.

2. Касьянов В.Н., Евстигнеев В.А. Графы в программировании: обработка, визуализация и применение. – СПб.: БХВ-Петербург, 2003. – 1104с.

3. Эндрюс Г.Р. Основы многопоточного, параллельного и распределенного программирования.: пер.с англ. – М.:Издательский дом «Вильямс», 2003. – 512с.

4. Шичкина Ю.А. Сокращение высоты информационного графа параллельного алгоритма. Научно-технические ведомости СПбГПУ. Информатика. Телекоммуникации. Управление. -2009. – №3(80). – с.148-152.