Разработка и анализ параллельных алгоритмов поиска максимального паросочетания в двудольном графе

А.С. Кириллов, А.Е. Шухман

Оренбургский государственный университет
alexey.s.kirillov@gmail.com, alex58@mail.ru

В настоящее время большое развитие получают технологии высокопроизводительных вычислений (High performance computing) и, в частности, параллельное программирование. С повышением вычислительных мощностей компьютерной техники появляется возможность решать практические задачи большой размерности. Одной из таких задач является задача поиска максимального паросочетания в двудольном графе. Данная задача находит очень большое применение на практике [1] в таких науках как математика, физика и химия.

Мы рассмотрели некоторые последовательные алгоритмы поиска максимального паросочетания:

– алгоритм Форда-Фалкерсона,
– алгоритм Эдмондса-Карпа,
– алгоритм Куна,
– алгоритм Хопкрофта-Карпа,
– алгоритм Муча-Санковски.

Для реализации последовательной версии приложения поиска максимального паросочетания в двудольных графах был выбран алгоритм Куна по двум причинам [2]:

а) алгоритм Куна имеет приемлемую сложность (в случае использования поиска в ширину сложность алгоритма составляет $O(n^{2,5})$);

б) алгоритм Куна имеет достаточно простую реализацию, что облегчает построение параллельных версий приложения.

На основании последовательной версии приложения были реализованы параллельные алгоритмы поиска максимального паросочетания с использованием парадигмы «Хозяин-Слуга» и методологии «Разделяй и властвуй».

Схема работы алгоритма при использовании парадигмы «Хозяин-Слуга» представлена в соответствии с рисунком 1.

Схема работы алгоритма при использовании методологии «Разделяй и властвуй» представлена в соответствии с рисунком 2.

Из описания алгоритмов можно заметить что:

а) при использовании парадигмы «Хозяин-Слуга» происходит очень много синхронизаций между процессами «Слугами» и процессом «Хозяином», что будет очень сильно снижать производительность приложения за счет накладных расходов на подготовку запроса и его последующую пересылку по сети;

б) при использовании методологии «Разделяй и властвуй» происходит небольшое число синхронизаций между всеми процессами (всего три раза за каждый шаг цикла и один раз в конце выполнения приложения), что не будет снижать производительность приложения за счет накладных расходов по пересылке запросов, по сравнению с первым вариантом параллельного приложения.

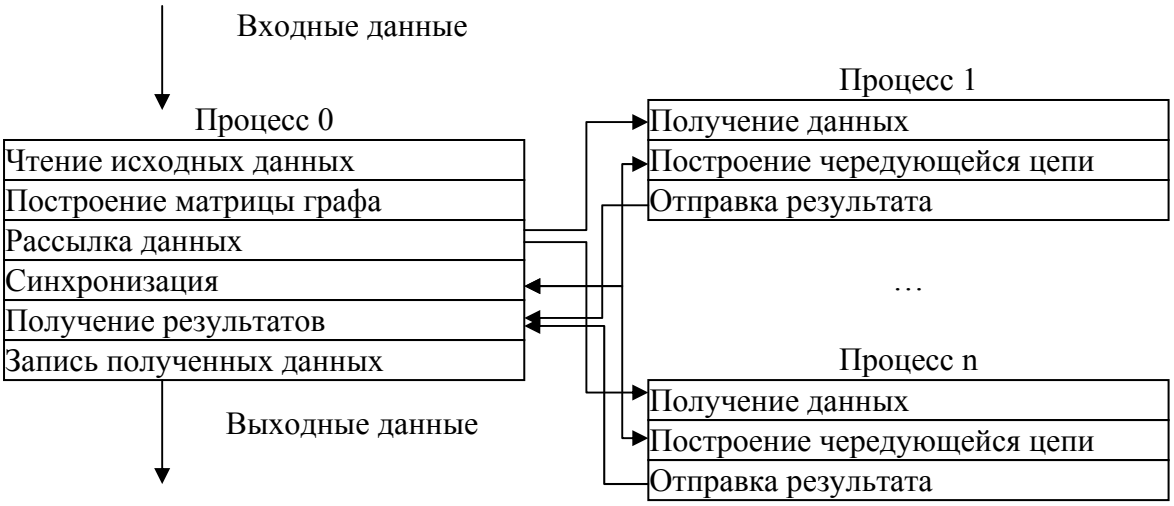

Рисунок 1. Схема работы алгоритма при использовании парадигмы «Хозяин-Слуга».

Рисунок 2. Схема работы алгоритма при использовании методологии0 «Разделяй и властвуй».

Для проведения вычислительного эксперимента и сбора экспериментальных данных был использован вычислительный кластер Государственного образовательного учреждения высшего профессионально образования «Оренбургский государственный университет».

Кластер ОГУ состоит из 6 вычислительных узлов и 1 головного узла, объединенных коммуникационной сетью Infiniband, скорость передачи которой составляет 10

Гбит/с, и управляющей сетью Gigabit Ethernet, скорость передачи которой составляет 1 Гбит/с. Пиковая производительность кластера составляет 89.6 GFlops (или 89.6 миллиардов операций в секунду).

Головной узел включает 4 процессора Intel Xeon с тактовой частотой 3.2 ГГц, 2 Гигабайта оперативной памяти и жесткий диск объемом 320 Гбайт (+320 Гбайт в резерве).

Каждый вычислительный узел включает 2 процессора Intel Xeon с тактовой частотой 3.2 ГГц, 2 Гигабайта оперативной памяти и жесткий диск объемом 80 Гбайт.

Для сбора экспериментальных данных был проведен вычислительный эксперимент, который выполнялся в несколько этапов.

Этап 1. Генерация произвольных исходных данных с количеством вершин в левой и правой долях равными: 500, 1000, 5000 и 10000. Причем для каждого размера исходных данных было сформировано по пять наборов исходных данных (всего двадцать наборов исходных данных). При этом количество ребер в графе было в пределах от $5n$ до $10n$, где n –количество вершин в левой доле графа.

Этап 2. Запуск последовательной версии приложения с каждым набором данных с замером времени работы приложения и последующая проверка корректности найденного решения.

Этап 3. Запуск первой версии параллельной версии приложения с каждым набором данных на 2, 4, 6 и 8 процессорах кластера ОГУ с замером времени работы приложения и последующая проверка корректности найденного решения.

Этап 4. Запуск второй версии параллельной версии приложения с каждым набором данных на 2, 4, 6 и 8 процессорах кластера ОГУ с замером времени работы приложения и последующая проверка корректности найденного решения.

Этап 5. После сбора данных о времени работы каждого приложения с каждым набором данных вычисляется среднее время работы для каждого размера исходных данных.

При работе первой параллельной версии приложения было отмечено очень большое время работы. Вследствие этого было принято решение для первой параллельной версии приложения не производить вычислительный эксперимент для количества вершин больше 1000.

Результаты сбора данных представлены в Таблице 1.

Как видно из Таблицы 1 первая параллельная версия приложения показала очень плохие результаты. Время работы первой параллельной версии во много раз превышает время работы последовательной версии приложения на тех же наборе исходных данных. Это обусловлено большими накладными расходами на синхронизацию данных между процессами, причем с увеличением количества процессов накладные расходы на синхронизацию процессов так же возрастают, и как следствие возрастает и общее время работы параллельного приложения.

В то же время, вторая параллельная версия показала время работы во много раз меньше времени работы последовательной версии приложения на тех же наборе исходных данных. Связано это с тем, что алгоритмическая сложность алгоритма поиска максимального паросочетания является кубической $O(n^3)$, то есть при уменьшении размера исходных данных в k раз алгоритмическая сложность уменьшается в k^3 раз. Кроме того, вторая параллельная версия приложения основывается на разбиении исходной области поиска на k непересекающихся областей, где k – количество процессоров.

Таблица 1. Время работы приложений в зависимости от количества вершин и количества процессоров

Количество вершин	Время работы приложения (сек.)								
	Последовательная версия	Первая параллельная версия				Вторая параллельная версия			
	Количество процессоров								
	1	2	4	6	8	2	4	6	8
500	0,2271	8,3495	5,8626	6,4160	23,775	0,0547	0,0681	0,1151	1,1018
1000	1,7143	64,548	43,372	38,245	197,03	0,3178	0,1682	0,2030	1,1730
5000	212,28	-	-	-	-	28,966	6,3203	4,1963	4,7328
10000	1694,0	-	-	-	-	221,00	41,322	19,877	16,659

Проведем экспериментальную оценку сложности алгоритма. Представим сложность алгоритма в общем виде степенной функции $y=a\cdot x^b$, получим:

$$T(n) = a \cdot n^b \tag{1}$$

где T(n) – время работы приложения;
n – размер исходных данных;
a, b – неизвестные коэффициенты.

Значения переменных $T(n)$ и n представлены в Таблице 2 и взяты на основании данных вычислительного эксперимента из Таблицы 1 при количестве процессоров равном 1 и 8 для $T_1(n)$ и $T_2(n)$ соответственно.

Таблица 2 – Значения переменных $T(n)$ и n

Размер исходных данных (n)	500	1000	5000	10000
Время работы последовательной версии приложения ($T_1(n)$)	0,2271	1,7143	212,28	1694
Время работы второй параллельной версии приложения на 8 процессорах ($T_2(n)$)	1,1018	1,173	4,7328	16,659

Для нахождения неизвестных коэффициентов a, b прологарифмируем левую и правую части формулы (1), получим:

$$\ln(T(n)) = \ln a + b \ln n \tag{2}$$

Заменим в формуле (2) $\ln n$, $\ln(T(n))$ и $\ln a$ на X, Y и A соответственно, получим:

$$Y = A + b \cdot X \tag{3}$$

Получили линейную зависимость, наилучшие коэффициенты в которой можно найти методом наименьших квадратов.

В результате получаем:

$$T_1(n) = 2 \times 10^{-9} \cdot n^{2,981}$$
$$T_2(n) = 3,1 \times 10^{-3} \cdot n^{0,898} \qquad (4)$$

Как видно из формул (4), мы получили, что экспериментальная алгоритмическая сложность параллельного алгоритма с использованием методологии «Разделяй и властвуй» поиска максимального паросочетания в двудольном графе имеет практически линейную алгоритмическую сложность.

Литература

1. Ловас, Л. Прикладные задачи теории графов [Текст] : Теория паросочетаний в математике, физике, химии : [Пер. с англ.] / Л. Ловас, М. Пламмер — М.: Мир, 1998. — 653 с, ил. — ISBN 5-03-002517-0
2. MAXimal [Электронный ресурс] / Алгоритм Куна нахождения наибольшего паросочетания в двудольном графе — Режим доступа: http://e-maxx.ru/algo/kuhn_matching , свободный. — Загл. с экрана. — Яз. рус.