Реализация последовательного и параллельного алгоритма ленточного умножения матриц.

Выполнил Петр Новоселов, студент группы 11-706.

Результатом перемножения матриц A и B является матрица C, каждый элемент которой есть скалярное произведение соответствующих строк матрицы A и столбцов матрицы B.

Задача состоит в получении результата перемножения двух случайно заданных квадратных матриц.

Для решения используются последовательный и параллельный алгоритмы, результаты которых нужно сравнить. Параллельный алгоритм разрабатывается на основе последовательного, с целью ускорения выполнения процесса умножения матриц больших размеров. Существует несколько методов распараллеливания алгоритма перемножения матриц. В этой работе будет реализован так называемый ленточный алгоритм.

Последовательный алгоритм

Этот алгоритм является итеративным и ориентирован на последовательное вычисление строк матрицы C. Предполагается выполнение n•n•n операций умножения и столько же операций сложения элементов исходных матриц. Количество выполненных операций имеет порядок $O(n^3)$.

Поскольку каждый элемент результирующей матрицы есть скалярное произведение строки и столбца исходных матриц, то для вычисления всех элементов матрицы C размером n*n необходимо выполнить $(n^2)*(2n-1)$ скалярных операций и затратить время $T1 = (n^2)*(2n-1)*t$, где t есть время выполнения одной элементарной скалярной операции.

Параллельный алгоритм

Алгоритм представляет собой итерационную процедуру, количество итераций которой совпадает с числом подзадач.

- На каждой итерации алгоритма каждая подзадача содержит по одинаковому количеству строк матрицы A и матрицы B.
- При выполнении итерации проводится скалярное умножение содержащихся в подзадачах строк, что приводит к получению соответствующих элементов результирующей матрицы C.
- По завершении вычислений в конце каждой итерации столбцы матрицы B должны быть переданы между подзадачами с тем, чтобы в каждой подзадаче оказались новые столбцы матрицы B и могли быть вычислены новые элементы матрицы C.

- По завершении итераций строки собираются в единую матрицу C.

При этом данная передача столбцов между подзадачами должна быть организована таким образом, чтобы после завершения итераций алгоритма в каждой подзадаче последовательно оказались все столбцы матрицы B.

Выделенные базовые подзадачи характеризуются одинаковой вычислительной трудоемкостью и равным объемом передаваемых данных. Когда размер матриц n оказывается больше, чем число процессоров p, базовые подзадачи можно укрупнить, объединив в рамках одной подзадачи несколько соседних строк и столбцов перемножаемых матриц. В этом случае исходная матрица A разбивается на ряд горизонтальных полос, а матрица B представляется в виде набора вертикальных полос. Размер полос при этом следует выбрать равным k=n/p (в предположении, что n кратно p), что позволит по-прежнему обеспечить равномерность распределения вычислительной нагрузки по процессорам, составляющим многопроцессорную вычислительную систему.

Анализ эффективности параллельного подхода

Примем за T_s - время работы последовательного алгоритма, за T_p - время работы параллельного алгоритма, за S - ускорение, за E - эффективность, за p – количество процессоров.

При применении последовательного алгоритма перемножения матриц число шагов имеет порядок $O(n^3)$.

Для параллельного алгоритма на каждой итерации каждый процессор выполняет умножение имеющихся на процессоре полос исходных матриц A и B (размерность каждой полосы составляет n/p, следовательно, общее количество выполняемых при этом умножении операций равно n^3/p^2). Поскольку число итераций алгоритма совпадает с количеством процессоров, сложность параллельного алгоритма без учета затрат на передачу данных может быть определена при помощи выражения:

$T_p = (n^3/p^2) * p = n^3/p.$

Показатели ускорения и эффективности данного параллельного алгоритма можно записать как:

$S = T_s/T_p = n^3 / (n^3 / p) = p$

$E = n^3 / [(n^3 / p) * p] = 1$

Полученные показатели являются идеальными. На практике же мы обычно получаем относительные показатели, которые с идеальными не совпадают из-за затрат на выполнение операций передачи данных между процессорами, служебных затарт и пр.

Оценим реальные показатели с учётом затрат на выполнение операций передачи данных между процессорами. С учетом числа и длительности выполняемых операций

время выполнения вычислений параллельного алгоритма может быть оценено следующим образом:

$T_p(calc) = [n^2 / p] * [2 n - 1] * t,$

где t - время выполнения одной элементарной скалярной операции.

Для оценки коммуникационной сложности параллельных вычислений будем предполагать, что все операции передачи данных между процессорами в ходе одной итерации алгоритма могут быть выполнены параллельно. Объем передаваемых данных между процессорами определяется размером полос и составляет n/p строк или столбцов длины *n*. Общее количество параллельных операций передачи сообщений на единицу меньше числа итераций алгоритма (на последней итерации передача данных не является обязательной). Тем самым, оценка трудоемкости выполняемых операций передачи данных может быть определена как:

$T_p (comm) = (p - 1) * (a + w * n * (n / p) / b),$

где a - латентность, b - пропускная способность сети передачи данных, а w - размер элемента матрицы в байтах. Тогда

$T_p = (n^2 / p) * (2 n - 1) * t + (p - 1) * (a + w * n * (n / p) / b).$

Результаты тестирования

Порядок матрицы	Последовательный алгоритм	Параллельный алгоритм, 2 процесса		Параллельный алгоритм, 4 процесса	
	Время выполнения	Время выполнения	Ускорение	Время выполнения	Ускорение
100	0,03	0,017	1,81392	0,02	1,90067
200	0,22	0,13	1,74627	0,63	0,34828
300	0,73	0,28	2,64382	0,69	1,05927
400	1,72	0,54	3,17026	1,26	1,36459
500	3,32	0,96	3,48165	1,88	1,76962
600	5,74	1,62	3,55103	2,90	1,97707

700	9,19	2,58	3,56380	4,41	2,08321
800	13,69	3,81	3,58699	6,32	2,16517
900	19,49	5,40	3,61055	8,86	2,19889
1000	26,71	7,35	3,63260	11,77	2,26963