МОДЕЛИ ОТОБРАЖЕНИЯ МНОГОШАГОВЫХ АЛГОРИТМОВ НА ПАРАЛЛЕЛЬНЫЕ ВЫЧИСЛИТЕЛЬНЫЕ СИСТЕМЫ С ТОПОЛОГИЕЙ ГИПЕРКУБ

Дмитриева О.А.

Кафедра ПМИ ДонНТУ
dmitriv@r5.dgtu.donetsk.ua

Abstract

Dmitrieva O. Reflection model of multistep algorithms on parallel calculable systems with gipercub topology. *Multistep block methods for decision Cauchy problem for usual differential equations, that are competent on realization in calculable systems with parallel architecture, are introduce up. reflection of extra-exact multistep block algorithms on parallel calculable structures SIMD and MIMD types with gipercub topology is realized. The realization peculiarities multistep multipoint block methods on parallel structures with vary dimensions of processor fields are definite. Comparative descriptions of offered reflection methods and recommendation on selection of worked up algorithms for diverse types of solved systems of usual differential equalizations are given. Parallelizm estimations are got: acceleration and effectiveness for offered decision reflection methods.*

Введение

В последние годы значительно расширился круг работ по теории параллельных вычислений. Переосмысливаются традиционные подходы к построению новых численных алгоритмов, большая работа проводится по разработке новых подходов для решения широких классов сложных задач, в том числе и задач большой размерности. В первую очередь это объясняется тем, что производительность вычислительной системы в целом напрямую зависит от адекватности отображения структуры вычислительного алгоритма на схему межпроцессорных связей. В данной работе представлена технология, реализующая высокоточные многошаговые блочные алгоритмы [1] решения задачи Коши для систем обыкновенных дифференциальных уравнений (СОДУ) на параллельных вычислительных структурах SIMD (single instruction / multiple data) и MIMD (multiple instruction / multiple data) типов с топологией гиперкуб.

Топология гиперкуб выбрана в связи с тем, что другие схемы коммутации, определяемые пользователем, такие как сетка, тор, кольцо, цилиндр, относительно легко создаются из гиперкубической схемы связи. К тому же, основным преимуществом вычислительных систем с гиперкубической топологией является значительное сокращение времени, затрачиваемого на обмены, при этом эффективность сетевой топологии измеряется, в частности, числом шагов для передачи данных между наиболее удаленными процессорами в системе. Для гиперкуба максимальное расстояние (число шагов) между процессорами совпадает с размерностью куба. Таким образом, никакая другая топология соединения процессоров не может сравниться с гиперкубом по эффективности. Большое число соединений в гиперкубе создает самую высокую пропускную способность межпроцессорных соединений по сравнению с любой другой сетевой схемой. Такое количество путей передачи данных позволяют передавать данные с очень высокой скоростью.

Реализация многошагового многоточечного блочного метода в SIMD-системах с топологией гиперкуб

Оптимальность отображения на конкретную архитектуру ЭВМ зависит от многих факторов, в том числе от размерности задачи, от числа процессорных элементов, от скорости обмена данными, от производительности процессоров и т.д. Для эффективной реализации разработанных алгоритмов выбирается стратегия распределения данных по процессорам, определяются свойства (степень параллелизма, масштабируемость, время работы) основных его частей.

В общем случае алгоритм решения задачи Коши для СОДУ

$$\frac{dx}{dt} = f(t,x),\ x(t_0) = x_0 \qquad (1)$$

многошаговым многоточечным блочным методом [2]

$$\frac{u_{n,i} - u_{n,0}}{i\tau} = \sum_{j=1}^{k} b_{i,j} F_{n-1,j} + \sum_{j=1}^{k} a_{i,j} F_{n,j},\ i = \overline{1,k},\ n = 1,2,\ldots N \qquad (2)$$

где $F_{n,j} = f(t_n + j\tau, u(t_n + j\tau))$

разбивается на несколько подзадач, которые будут совершенно идентичными для каждого вычисляемого блока:

1. вычисление начальных приближений в точках нового блока;
2. вычисление правых частей функции для вновь получившихся значений;
3. подготовка и проведение итерационного процесса в блоке.

Первая подзадача выполняется для каждого блока только один раз и элементарно распараллеливается, т.к. начальные приближения значений для каждой точки вычисляются независимо, а время вычисления одинаково. Относительное время работы этой части алгоритма на параллельном компьютере будет меньше времени, затрачиваемого на эти же операции последовательной ЭВМ примерно во столько раз, сколько процессорных элементов участвует в вычислительном процессе, при условии, что за каждой точкой блока закреплен «свой» процессорный элемент.

Вторая подзадача состоит из обменов вновь полученными значениями, т.е. каждый процессорный элемент должен получить информацию от всех оставшихся, в которых были вычислены значения элементов вектора начального приближения для блока и, в то же время, переслать всем процессорным элементам «свое» значение элемента вектора. Для реализации такого способа обмена в работе предлагается подход, при котором для каждого узла его ближайшим соседом по гиперкубу является узел, чей номер в двоичной записи отличается на один бит от номера данного узла. Номер канала, который соединяет два соседних узла, равен увеличенному на единицу номеру битовой позиции несовпадающего бита. Алгоритм обмена по этому типу может быть проиллюстрирован с помощью упрощенных схем, представленных на рис. 1.

После того, как будет осуществлена рассылка «все-всем», необходимо вычислить правые части системы уравнений в каждой точке блока. Относительное время работы этой части алгоритма сильно варьируется в зависимости от однородности правых частей системы обыкновенных дифференциальных уравнений и их сложности.

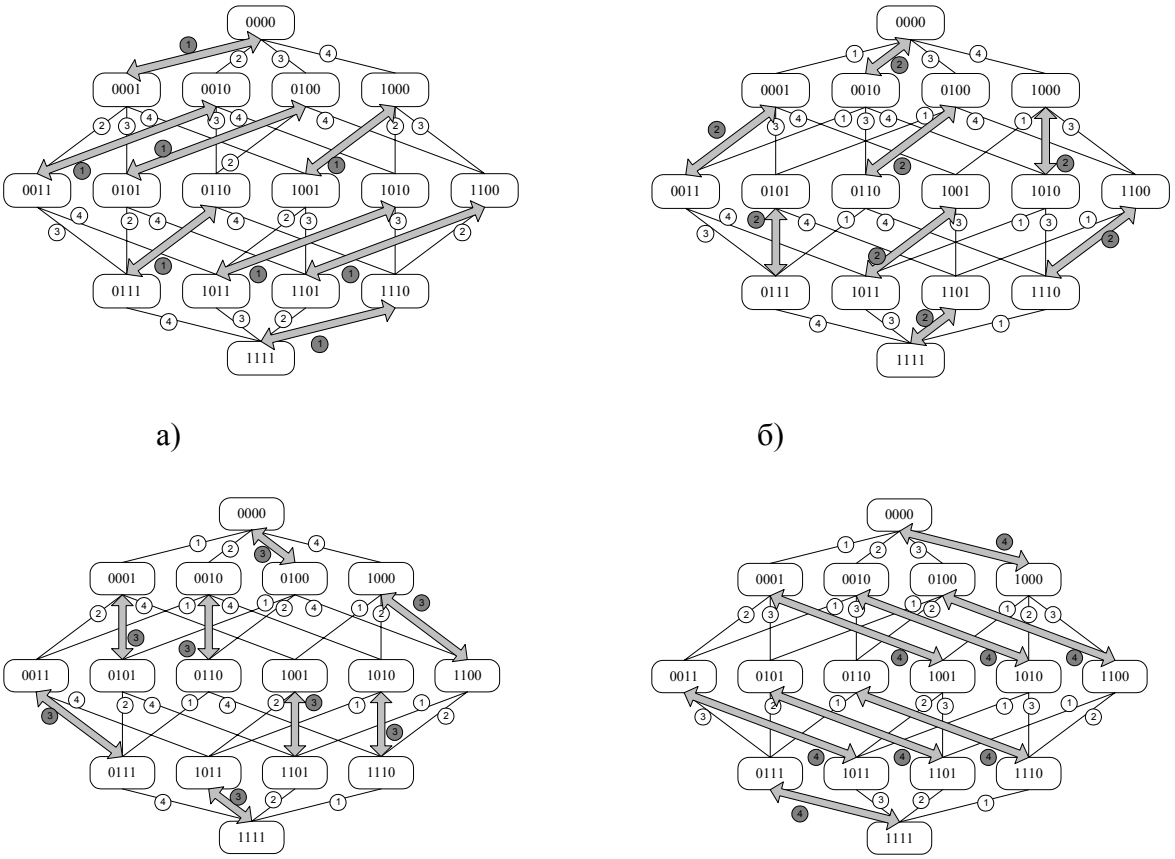
Последняя подзадача является самой трудоемкой и, можно сказать, включает в себя две уже описанные подзадачи, причем несколько раз циклически повторенные. К тому же, каждый следующий цикл требует некоторого переупорядочения данных для подготовки систолического умножения матрицы на вектор, что несет с собой дополнительные временные затраты. Если говорить о времени работы этой части алгоритма, то оно составляет 85-90% от общего. Тем не менее, эту подзадачу можно

значительно сократить, если достижение предельной точности $O(\tau^{2k})$ многошаговым многоточечным блочным методом не требуется. Если для решения системы достаточен, например, порядок погрешности $O(\tau^{k+1})$ - именно такой порядок точности дают первые две подзадачи, то вычисления можно прекратить, не переходя к подзадаче 3.

Исходя из поставленных подзадач, определяются необходимые коммуникационные примитивы [3]:

1. умножение матрицы на вектор. Для этого достаточно «размножить» значения каждого элемента вектора *k* раз (в соответствии с размерностью блока);

2. обмен по типу «все-всем», операции вычисления суммы путем комбинирования операций сдвига и сдваивания произведений правых частей и вспомогательных коэффициентов, распределенных по процессорам.



а) б)

в) г)

Рисунок 1 - Схема организации обменов «все-всем» в гиперкубе

Модели и технология отображения на SIMD-системы

При разработке коммуникационных примитивов была использована модель вычислений [4], представленная последовательностью супершагов, для которой время передачи данных между процессорами выражается в виде

$$T_{пер\_данных} = \sigma + \frac{m}{\tau}, \qquad (3)$$

где σ - время задержки для подготовительной работы (измеряется в количестве операций с плавающей точкой, которые можно выполнить за время этой задержки);

τ - скорость передачи (время выполнения одной операции с плавающей точкой);

m – число слов в передаваемом такте. Как оказалось, такая сравнительно простая модель позволяет хорошо описывать реально наблюдаемые величины на компьютерах типа MasPar, Intel Paragon, FPS(T).

В качестве SIMD-системы использовалась разработка фирмы FPS серии T в области архитектур с массовым параллелизмом. Конфигурация системы представляет собой n- мерный гиперкуб – структуру, в которой каждый узел связи связан с n ближайшими соседями. К системе присоединяется внешняя машина (front-end computer), которая обеспечивает возможности по разработке программ и управлению системой. Количество процессорных элементов в системе равно *p*. Рассматриваются два способа реализации вычислений.

Для первого способа под каждое уравнение системы будет выделяться k^2 процессорных элементов, т.е., каждую точку блока будут рассчитывать *k* микропроцессоров. При втором способе размещения данных за каждой точкой блока закрепляется только один процессор. Разумеется, общее число процессорных элементов в системе не должно быть меньше, чем квадрат размерности блока, т.е., справедливо соотношение $p \geq k^2$. Размерность системы обыкновенных дифференциальных уравнений может быть произвольной, в том числе и превосходить имеющееся количество процессорных элементов. В этом случае для удобства можно принять, что размерность системы *m* кратна отношениям p/k^2 для первого алгоритма и p/k для второго алгоритма, другими словами $m = \gamma \frac{p}{k^2}$ или $m = \gamma \frac{p}{k}$ (предположение, необходимое только для удобства рассмотрения).

Тогда для первого алгоритма величина γ или $m/p/k^2$ - количество «тактов», за которые можно рассчитать всю систему для блока из *k* точек. Для вычисления начального приближения значений функции в процессорном поле размерностью $k \times k$ элементов, для итерирования и подготовки вычислений в новом блоке потребуется время, определяемое как

$$T_{гипер\_k\times k} = k[\frac{mk^2}{p}(2t_{умн} + \log_2 pt_{сдв} + (\log_2 k + 1)t_{сл}) + \log_2 pt_{сдв} + \frac{m}{k}t_f)] +$$
$$t_{умн} + \log_2 pt_{сдв} + (\log_2 k + 1)t_{сл}.$$

где $t_{сл}, t_{умн}, t_{сдв}$ - времена, необходимые на реализацию операций сложения, умножения и сдвига соответственно, t_f – время, необходимое на вычисление правых частей уравнений системы (1).

Для реальной оценки времени, необходимого при расчете значений необходимо учесть, что время пересылки данных между процессорными элементами соизмеримо со временем выполнения арифметических операций с плавающей точкой [4]. Поскольку для нормального функционирования в алгоритме не предусмотрено обращение к внешней машине, можно принять, что время выполнения одной операции соизмеримо с временами $t_{сл} \approx t_{умн} \approx t_{сдв}$. Тогда время, необходимое для расчета значений в одном блоке можно будет оценить как

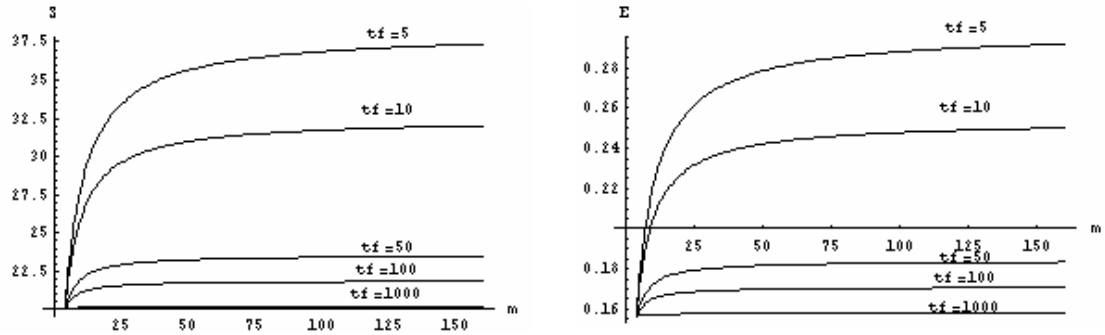
$$T_{гипер\_k\times k} \approx k[\frac{mk^2}{p}(\log_2 p + \log_2 k + 3)t_{оп} + \log_2 pt_{оп} + \frac{m}{k}t_f] + \quad (4)$$
$$(\log_2 p + \log_2 k + 2)t_{оп}.$$

Для такого способа размещения данных характеристики параллелизма представляются на рис. 2 с помощью графиков.

Для второго способа расположения данных, а именно, когда для расчета каждой точки блока выделяется только один процессор, а на каждое уравнение системы *k* процессоров, общее время будет определяться как

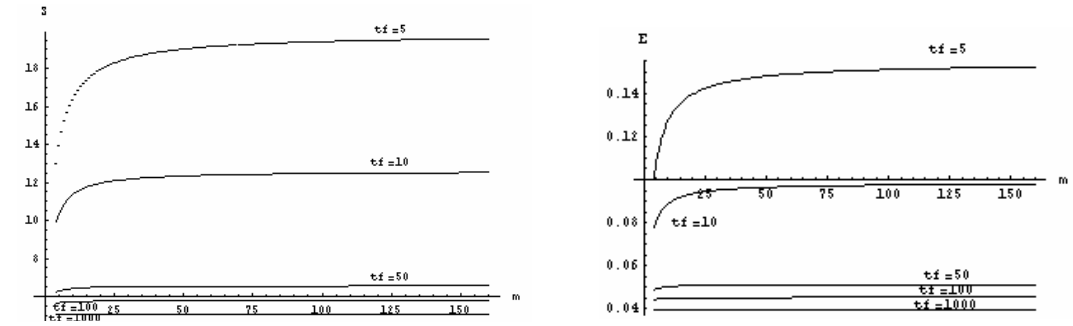
$$T_{гипер\_k} = k[\frac{2mk}{p}(k+1)t_{on} + log_2\ pt_{on} + mt_f\ ] + (2(k+1) + log_2\ p)t_{on}. \qquad (5)$$

В качестве основных характеристик параллелизма для первого способа отображения при фиксированной размерности блока и варьируемой размерности системы на рис. 3 приведены графики ускорения и эффективности для различных времен вычисления правых частей.



а) ускорение б) эффективность

Рисунок 2 - Характеристики параллелизма блочных методов при реализации на структурах типа FPS («глубокое» распараллеливание)



а) ускорение б) эффективность

Рисунок 3 - Характеристики параллелизма блочных методов при реализации на структурах типа FPS («мелкое» распараллеливание)

Для способов коммутации 2D-, 3D-тор и гиперкуб приемлемые показатели ускорения и эффективности блочных алгоритмов достигаются только для случая линейных систем в связи с невозможностью выполнения разнотипных операций. Значения предельных показателей ускорения (S) и эффективности (E) приведены в таблице 1.

Таблица 1 - Основные показатели параллелизма реализованных методов для SIMD систем

	Топологическое соединение	Нелинейные системы		Линейные системы	
		S	E	S	E
1.	1D- тор (p≥k)	~k	~1	~k	~1
2.	1D- тор (p≥m)	~k	~k/m (m≥k)	~m	~1
3.	2D- тор (p≥k×k)	~k	~1/k	~k×k/$\log_2$k	~1/$\log_2$k
4.	2D- тор (p≥m×k)	~k	~1/m	~m×k/$\log_2$m	~1/$\log_2$m
5.	3D- тор (p≥k×k×m)	~k	~1/(m×k)	~k×k×m/$\log_2$(k+m)	~1/$\log_2$(k+m)
6.	Гиперкуб (p≥k×k)	~k	~1/k	~k×k/$\log_2$(k-1)	~1/$\log_2$(k-1)

Организация отображения решения систем обыкновенных дифференциальных уравнений на параллельные вычислительные MIMD структуры с топологией «гиперкуб» осуществляется на основе Intel iPSC.

Все процессорные узлы идентичны и соединены двусторонними связями по топологии гиперкуба. Коммуникационная среда позволяет с помощью программ пользователя задавать и другие топологии системы, такие как сетки, кольца, деревья.

Соотношение между временами арифметических операций и временами пересылки данных в этой системе представляется как

$$t_{сл} \approx t_{умн} \approx t_{оп}, \quad t_{сдв} \approx \frac{3*10^{-3}}{3*10^{-4}} \approx 10 t_{оп} \tag{6}$$

Полученная оценка для времени сдвигов считается «верхней», и можно с уверенностью сказать, что время обменов не превысит этой величины. В силу этих оценок можно выразить время, необходимое для вычисления значений в одном блоке при использовании схемы расположения значений, приведенной в [4] следующим образом:

$$T_{k\times k} \approx k\,[\frac{mk^2}{p}(10(k-1)t_{оп} + (\log_2 k + 3)t_{оп}) + \log_2 p t_{оп} + \\ + \frac{mk}{p} t_f\,] + 10\log_2 p t_{оп} + (\log_2 k + 2)t_{оп}. \tag{7}$$

Реально влияющими на скорость обработки будут времена

$$T_{k\times k} \approx \frac{10mk^4}{p} t_{оп} + \frac{mk^2}{p} t_f \tag{8}$$

Характеристики параллелизма для такой схемы приведены на рис. 4.

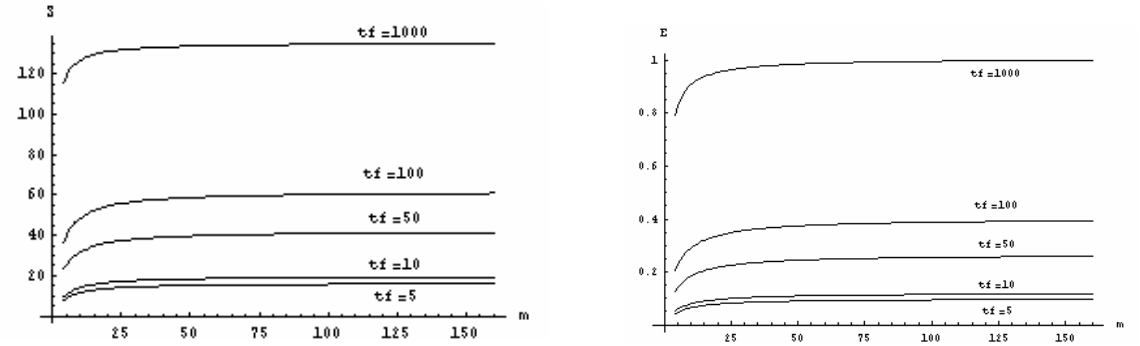
При втором способе расположения данных время вычисления значений для одного блока будет определяться как

$$T_k \approx k[\frac{2mk}{p}(k+1)t_{оп} + 10\log_2 p t_{оп} + \frac{mk}{p} t_f\,] + \\ + 2(k+1)t_{оп} + 10\log_2 p t_{оп}. \tag{9}$$

При том же соотношении времен арифметических операций и обменов реально влиять на время решения будут величины

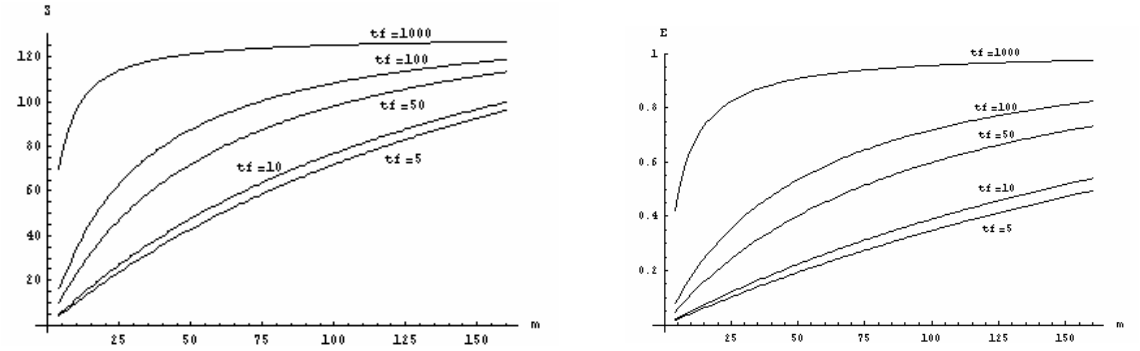
$$T_k \approx \frac{2mk^3}{p} t_{оп} + \frac{mk^2}{p} t_f \tag{10}$$

Характеристики параллелизма для такой схемы приведены на рис. 5. Из рис. 4 и 5 видно, что если расчет будет вестись для системы обыкновенных дифференциальных уравнений, характеризующейся высокой трудоемкостью вычисления правых частей, то первыми слагаемыми в формулах (8) и (10) можно будет пренебречь, и порядок расположения данных в вычислительных модулях не будет влиять на быстродействие многопроцессорной вычислительной системы, в противном случае (например, для однородных систем обыкновенных дифференциальных уравнений), вторая схема расположения данных будет предпочтительнее, поскольку в этом случае сокращение количества обменов позволит значительно ускорить быстродействие системы.



а) ускорение б) эффективность

Рисунок 4 - Характеристики параллелизма блочных методов при реализации на структурах Intel iPSC («глубокое» распараллеливание)



а) ускорение б) эффективность

Рисунок 5 - Характеристики параллелизма блочных методов при реализации на структурах Intel iPSC («мелкое» распараллеливание).

Заключение

В результате отображения высокоточных многошаговых блочных алгоритмов на параллельные вычислительные структуры SIMD и MIMD типов с топологией гиперкуб определены особенности реализации многошаговых многоточечных блочных методов на топологических структурах с варьируемыми размерностями процессорных полей. Даны сравнительные характеристики предложенных способов отображения и рекомендации по выбору разработанных алгоритмов для различных типов решаемых систем обыкновенных дифференциальных уравнений. Получены оценки параллелизма: ускорение и эффективность для предложенных способов отображения решения. Показано, что если расчет будет проводиться для систем со сложными правыми

частями (время вычисления правых частей будет доминировать), то трудоемкость обоих способов отображения будет практически одинаковой. Если правые части системы будут тривиальными, то использование второго способа предпочтительнее. Определено, что «узким» местом решения систем обыкновенных дифференциальных уравнений (за исключением однородных и линейных) на всех параллельных вычислительных структурах типа SIMD является невозможность одновременного вычисления значений правых частей. Поэтому использование MIMD – систем для решения систем обыкновенных дифференциальных уравнений является более предпочтительным вариантом. Доказано, что эффективность параллельных вычислений в MIMD – системах достигает своего максимума, если удается сократить время выполнения обменов, причем это сокращение может достигаться не только за счет повышения пропускной способности линков (аппаратно), но и в результате целенаправленного программирования решаемых задач, обеспечивающего повышение отношения времени обработки к времени передачи данных. Именно за счет целенаправленного программирования (использования систолических алгоритмов умножения матриц, оптимального размещения данных по процессорам) в предложенных способах отображения значительно сокращено количество обменов.

Эта задача становится актуальной еще и потому, что скорость обработки данных в микропроцессорах растет существенно быстрее увеличения пропускной способности интерфейсов. И, поскольку заранее, при написании и компиляции программы невозможно точно предсказать время исполнения обменов данными, очевидно, что сокращение количества обменов будет приводить к повышению эффективности вычислительных систем при решении задач большой размерности.

Литература

1. Фельдман Л.П., Дмитриева О.А. Эффективные методы распараллеливания численного решения задачи Коши для обыкновенных дифференциальных уравнений.// Математическое моделирование, том 13, № 7, 2001. – С.66-72.
2. Дмитриева О.А. Параллельные блочные многошаговые алгоритмы численного решения систем обыкновенных дифференциальных уравнений большой размерности. //Научные труды Донецкого государственного технического университета. Серия: Информатика, кибернетика и вычислительная техника (ИКВТ-2000), выпуск 15: - Донецк: ДонГТУ, 2000. – С. 53-58.
3. Дмитриева О.А. Анализ параллельных алгоритмов численного решения систем обыкновенных дифференциальных уравнений методами Адамса-Башфорта и Адамса-Моултона.// Математическое моделирование. – 2000. – Т. 12, № 5. - С. 81-86.
4. Feldman L., Dmitrieva O., Gerber S. Abbildung der blockartigen Algoritmen auf die Parallelrechnerarchitekture. 16 Symposium Simulationstechnik ASIM 2002, Rostock, 10.09 bis 13.09 2002. – Erlangen: Gruner Druck, 2002. – P. 359-364
5. Фельдман Л.П., Дмитриева О.А. Разработка и обоснование параллельных блочных методов решения обыкновенных дифференциальных уравнений на SIMD-структурах. //Научн. тр. Донецкого государственного технического университета. Серия: Проблемы моделирования и автоматизации проектирования динамических систем, выпуск 29. - Севастополь: «Вебер», 2001. - С. 70-79.

Поступила в редакційну колегію 28.12.2002