УДК 519.254 + 519.163

С.Н. Ефименко, В.С. Степашко

Основы рекуррентно-параллельных вычислений в комбинаторном алгоритме *COMBI* МГУА

Разработаны теоретические основы распараллеливания операций в комбинаторном алгоритме *COMBI* МГУА с рекуррентным вычислением параметров моделей. Представлена теоретическая оценка эффективности разработанной схемы. Проведено экспериментальное исследование эффективности применения предложенной схемы.

Theoretical grounds of operations paralleling in COMBI GMDH combinatorial algorithm with recurrent parameters estimation are developed. Theoretical estimate of constructed scheme efficiency is proposed. The results of experimental investigation are presented.

Розроблено теоретичні основи розпаралелювання операцій в комбінаторному алгоритмі *COMBI* МГУА з рекурентним обчисленням параметрів моделей. Представлено теоретичну оцінку ефективності розробленої схеми. Проведено експериментальне дослідження ефективності застосування запропонованої схеми.

Введение. Метод группового учета аргументов (МГУА) [1] как основной инструмент теории индуктивного моделирования принадлежит к самым современным методам вычислительного интеллекта и мягких вычислений. Этот метод, разработанный академиком А.Г. Ивахненко, – оригинальное и эффективное средство решения широкого спектра задач искусственного интеллекта, в том числе идентификации и прогнозирования, распознавания образов и кластеризации, интеллектуального анализа данных и поиска закономерностей.

Важным критерием эффективности программных средств, базирующихся на методах индуктивного моделирования, есть время получения результата моделирования. Наиболее эффективными средствами достижения высокой производительности таких программных продуктов считаются рекуррентные вычисления и распараллеливание операций.

Высокопродуктивные средства на основе как рекуррентных, так и параллельных вычислений разработаны и продемонстрировали свою эффективность [1–5]. Цель данной статьи – объединение двух видов операций в единую парадигму параллельно-рекуррентных вычислений для перехода на более высокий уровень эффективности программных средств индуктивного моделирования.

Задача структурно-параметрической идентификации

В общем случае задача структурно-параметрической идентификации состоит в формировании по данным выборки некоторого множества моделей различной структуры вида

$$\hat{y}_f = f(X, \hat{\theta}_f) \qquad (1)$$

и поиске оптимальной модели согласно условию

$$f^* = \arg\min_{f \in \Im} CR(y, f(X, \hat{\theta}_f)), \qquad (2)$$

где оценки параметров в (2) для каждой модели $f \in \Im$ есть решением еще одной экстремальной задачи вида

$$\hat{\theta}_f = \arg\min_{\theta_f \in R^{s_f}} QR(y, X, \theta_f, s_f), \qquad (3)$$

где s_f называется сложностью модели f и равняется числу ненулевых компонент в модели (2); QR – критерий качества решения задачи параметрической идентификации каждой отдельной модели, которая генерируется в задаче структурной идентификации, где модели сравниваются по критерию CR. При этом критерии QR и CR должны быть различными, поскольку две соответствующие задачи по своей природе разные: (2) есть задачей дискретной, а (3) – непрерывной оптимизации.

Комбинаторный алгоритм *COMBI* МГУА

Комбинаторный алгоритм *COMBI* [1], предназначенный для решения задачи (1–3), т.е. для поиска (путем полного перебора всех возможных вариантов) лучшей регрессионной модели, содержащей наиболее информативное подмножество входных переменных (регрессоров), состоит из таких основных блоков:

- преобразования данных в соответствии с выбранным базисным классом моделей, линейных по параметрам;

• формирования моделей различной сложности;

• вычисления значений внешних критериев качества и отбор лучших моделей;

• оценки качества полученных моделей.

В случае линейного объекта с m входами в процессе полного перебора сравниваются все возможные модели вида

$$\hat{y}_v = X_v \hat{\theta}_v,\ v = \overline{1,\ ...,\ 2^m - 1}, \qquad (4)$$

где десятичному числу v соответствует двоичное число d_v, единичные элементы которого указывают на включение в модель регрессоров с соответствующими номерами, а нулевые – на исключение. Соответственно, формирование структур частных моделей формализируется с помощью двоичного структурного вектора

$$d_v = \{d_1, d_2, ..., d_m\},\ d_k = \{0;1\},\ k = \overline{1,m}.$$

Изменение состояний вектора d можно организовать различными способами. Достаточно удобна для полного перебора схема изменения вектора d по принципу двоичного счетчика, в последний разряд которого добавляется единица. При этом существует однозначное соответствие между десятичным порядковым номером очередной модели и состоянием двоичного структурного вектора.

При переборе сложность частных моделей изменяется от единицы до m. Общее число вариантов при полном переборе составляет $2^m - 1$ различных структур, т.е. зависимость объема перебора от числа аргументов/регрессоров m – экспоненциальна. Это обстоятельство ставит самые высокие требования к организации вычислительных операций в таких алгоритмах с учетом их максимального ускорения.

В табл. 1 приведены оценки зависимости времени моделирования с помощью самого быстрого варианта комбинаторного алгоритма (без распараллеливания вычислений) от количества аргументов при полном переборе. В принципе целесообразно рассматривать возможность построения модели на компьютере с одним процессором за приемлемое время при m не более 30. При количестве аргументов больше 35 перебор всех вариантов за допустимое время на персональном компьютере становится нецелесообразным.

Задача повышения вычислительной эффективности комбинаторного алгоритма может быть решена с использованием как быстродействующих методов оценивания параметров, базирующихся на рекуррентных алгоритмах решения систем линейных алгебраических уравнений, так и с применением распараллеливания вычислений с помощью многопроцессорных кластерных систем.

Таблица 1. Время полного перебора

Количество аргументов	Количество моделей	Время моделирования
20	1 048 576	1 с
21	2 097 152	2 с
22	4 194 304	4 с
…	…	…
30	~ 1E + 9	~ 20 м
…	…	…
36	~7E + 10	~ 1 день
…	…	…
45	~4E + 13	~ 1 год

Рекуррентное оценивание параметров

Задача оценивания параметров – наиболее трудоемкая в комбинаторном алгоритме МГУА. Поскольку в случае моделей, линейных по параметрам, в этой задаче применяется метод наименьших квадратов (МНК), фактически речь идет об ускорении решения систем линейных уравнений.

Традиционным и эффективным рекуррентным методом есть метод окаймления [1, 2]. Идея этого алгоритма – модификации МНК – состоит в пошаговом рекуррентном вычислении и корректировании оценок параметров моделей возрастающей сложности с рекуррентным вычислением элементов обратной матрицы системы нормальных уравнений.

В [3] были также предложены эффективные рекуррентные модификации классических алгоритмов Гаусса и Грамма–Шмидта, причем рекуррентный вариант метода Гаусса оказался весьма полезным для применения в параллельном комбинаторном алгоритме, поэтому приведем здесь его краткое описание.

Вычислительная суть модификации алгоритма Гаусса состоит в следующем. На s-м ша-

ге ($s = \overline{1, m}$) нормальная матрица $H_s = X_s^T X_s$ (для известной формулы оценки параметров по МНК $\hat{\theta} = (X^T X)^{-1} X^T y$) размерности $s \times s$ сводится к верхнему треугольному виду вычислением элементов $h_{is}^s, i = \overline{2, s-1}$ и $h_{si}^s, i = \overline{2, s}$, в то время как элементы вложенной матрицы H_{s-1} размерности $(s-1) \times (s-1)$, сведенной к верхнему треугольному виду на прежнем шаге, остаются неизменными. Таким образом, на s-м шаге выполняется вычисление «окаймления» к элементам расширенной матрицы системы нормальных уравнений, вычисленным на предыдущем шаге, и соответствующего элемента $g_s = X_s^T y$ этой матрицы:

$$\left[\begin{array}{ccccccccc|c}
h_{11} & h_{12} & h_{13} & \dots & h_{1s\text{-}1} & \boldsymbol{h_{1s}} & \dots & h_{1m} & g_1 \\
h_{21} & h_{22} & h_{23} & \dots & h_{2s\text{-}1} & \boldsymbol{h_{2s}} & \dots & h_{2m} & g_2 \\
h_{31} & h_{32} & h_{33} & \dots & h_{3s\text{-}1} & \boldsymbol{h_{3s}} & \dots & h_{3m} & g_3 \\
\dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\
h_{s\text{-}1,1} & h_{s\text{-}1,2} & h_{s\text{-}1,3} & \dots & h_{s\text{-}1,s\text{-}1} & \boldsymbol{h_{s\text{-}1,s}} & \dots & h_{s\text{-}1,m} & g_{s\text{-}1} \\
\boldsymbol{h_{s1}} & \boldsymbol{h_{s2}} & \boldsymbol{h_{s3}} & \boldsymbol{\dots} & \boldsymbol{h_{s,s\text{-}1}} & \boldsymbol{h_{ss}} & \dots & h_{sm} & \boldsymbol{g_s} \\
\dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\
h_{m1} & h_{m2} & h_{m3} & \dots & h_{m,s\text{-}1} & h_{ms} & \dots & h_{mm} & g_m
\end{array}\right].$$

На рис. 1 в графическом виде представлено сравнение показателей трудоемкости (количества элементарных арифметических операций) вычисления параметров при включении в модель, содержащую $s-1$ аргументов, s-го аргумента. Такая зависимость трудоемкости пропорциональна второй степени сложности модели для любого рекуррентного алгоритма и третьей степени – для нерекуррентного.

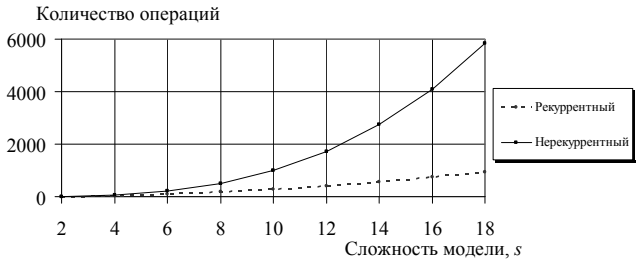

Рис. 1. Количество операций для рекуррентного и нерекуррентного алгоритмов

Распараллеливание вычислений

Метод оптимального распараллеливания операций в комбинаторном алгоритме МГУА с нерекуррентным оцениванием параметров на основе алгоритма генерации последовательно усложняемых структур, обеспечивающий одинаковое суммарное количество как моделей, так и оцениваемых параметров, приходящихся на каждый процессор, предложен в [4]. При этом последовательно строятся все модели с одним аргументом, затем с двумя, тремя и т.д., что гарантированно обеспечивает равномерную вычислительную нагрузку на любое заданное количество процессоров кластерного комплекса и дает возможность достаточно просто автоматизировать формирование задания на выполнение процесса распараллеливания.

С помощью тестового эксперимента (табл. 2) была подтверждена высокая эффективность разработанной схемы распараллеливания.

Таблица 2. Эффективность схемы распараллеливания комбинаторного алгоритма

	Количество аргументов				
	20	21	22	23	24
Количество процессоров	Эффективность, %				
1	100	100	100	100	100
2	100	99	100	100	99
4	100	100	100	98	99
8	100	99	100	98	99
16	99	98	98	100	100

Здесь эффективность определялась по формуле

$$E_k = \frac{T_1}{k \times T_k} \times 100\%, \quad k = 2^i, \quad i = \overline{0, 4}, \quad (5)$$

где T_1 – время выполнения алгоритма на одном процессоре, T_k – время выполнения на k процессорах.

Сравнение эффективности рекуррентных и параллельных вычислений

Сравнить между собой эффективность двух описанных способов увеличения производительности комбинаторного алгоритма МГУА позволяет эксперимент, описанный в [5]. Цель эксперимента состояла в сравнении времени выполнения полного перебора с использованием комбинаторного алгоритма:

- с нерекуррентным оцениванием параметров и распараллеливанием операций;
- с рекуррентным оцениванием параметров без распараллеливания вычислений.

На рис. 2 представлена зависимость времени выполнения программ от количества аргументов и процессоров кластерной системы. Последние (правые) столбцы диаграммы соответствуют алгоритму с рекуррентным оцениванием параметров на одном процессоре.

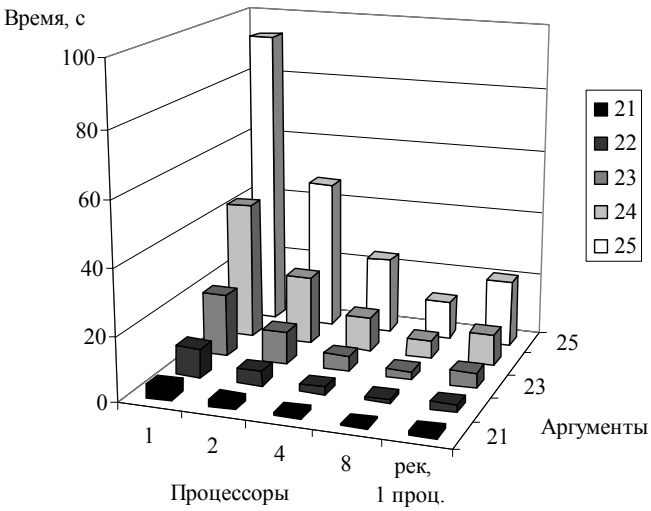

Рис. 2. Время выполнения комбинаторного алгоритма

Эффективность рекуррентного оценивания параметров определялась как отношение времени выполнения программ *COMBI* (с нерекуррентным и рекуррентным оцениванием параметров на одном процессоре) для количества аргументов от 21 до 25:

$$E_{rec} = \frac{T_{nonrec}}{T_{rec}}. \qquad (6)$$

Результат эксперимента, представленный на рис. 3, показывает рост значения E_{rec} с увеличением числа аргументов, а также позволяет сделать вывод о том, что полный перебор всех возможных моделей различной сложности по алгоритму *COMBI* с рекуррентным оцениванием параметров будет более эффективным (по времени выполнения) на персональных компьютерах с многоядерными процессорами (вплоть до четырех ядер), чем распараллеливание нерекуррентных вычислений на таких вычислительных устройствах. Очевидно, что в случае построения моделей на кластерных многопроцессорных комплексах использование распараллеливания операций представляется более предпочтительным.

Поскольку и рекуррентные, и параллельные вычисления в отдельности обеспечивают существенное уменьшение количества операций, затрачиваемых на процесс оценивания параметров, то сочетание этих двух мощных аппаратов позволяет получить эффект в виде недостижимого ранее повышения производительности алгоритмов МГУА.

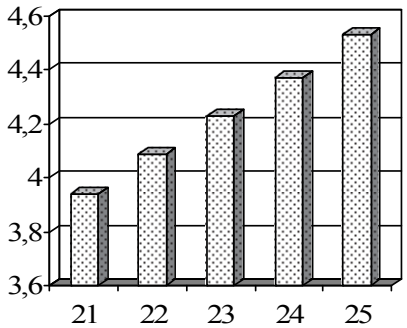

Рис. 3. Эффективность алгоритма с рекуррентными вычислениями

Комбинаторный алгоритм *COMBI* МГУА на основе рекуррентно-параллельных вычислений

Комбинаторный алгоритм имеет важную особенность, позволяющую сделать вывод о высокой эффективности его распараллеливания. Так, все блоки алгоритма могут выполняться каждым процессором автономно, без необходимости межпроцессорного взаимодействия. Главный процессор собирает результаты отдельных процессоров и отбирает лучшую модель.

Далее описывается разработанная схема распараллеливания комбинаторного алгоритма с рекуррентным вычислением параметров моделей с помощью модифицированного алгоритма Гаусса [6] решения систем линейных уравнений.

Схема использует генерацию двоичных чисел, отвечающих последовательным десятичным числам. Все возможные комбинации и последовательность двоичных структурных векторов для случая трех аргументов можно записать следующим образом: {0, 0, 0}; {0, 0, 1}; {0, 1, 0}; {0, 1, 1}; {1, 0, 0}; {1, 0, 1}; {1, 1, 0}; {1, 1, 1}.

Эту схему предлагается использовать для распараллеливания комбинаторного алгоритма на кластерные системы следующим образом.

Очевидно, что первая половина полной последовательности двоичных структурных векторов есть зеркальным отображением второй половины, в которой выполняется инвертирование элементов (нулевые элементы заменяются на единичные и наоборот). Так, для упомянутого ранее случая трех аргументов первому вектору {0, 0, 0} отвечает последний {1, 1, 1}, второму {0, 0, 1} – предпоследний {1, 1, 0} и т.д. Такая закономерность сохраняется для случая любой сложности моделей (общего количества аргументов) и позволяет решить задачу равномерного распределения количества моделей, а также обеспечить одинаковое суммарное количество аргументов на заданное количество процессоров кластерной системы.

Покажем, как можно распараллелить задачу для случая полного перебора из трех аргументов на кластерной системе, состоящей из двух процессоров. Получим модели с двоичными структурными векторами {0, 0, 0}; {0, 0, 1}; {1, 1, 0}; {1, 1, 1}, которые строит первый процессор (четыре модели с общим количеством аргументов, равным шести) и модели со структурными векторами {0, 1, 0}; {0, 1, 1}; {1, 0, 0}; {1, 0, 1}, строящие второй процессор (также получаем четыре модели с общим количеством аргументов, равным шести).

Запишем алгоритм нахождения последовательного набора двоичных структурных векторов при количестве аргументов, равном m, для κ-го процессора кластерной системы ($k = \overline{1, K}$) в виде последовательности шагов.

Ш а г 1. По формуле $P(m) = \sum_{s=1}^{m} C_m^s = 2^m$ вычисляем общее количество моделей, строящихся комбинаторным алгоритмом.

Ш а г 2. Находим количество моделей, которые будет строить каждый процессор: $P_k(m) = P(m)/K$. В случае если $P(m)$ не делится нацело на K, на первый процессор будет приходиться повышенная вычислительная нагрузка, отличающаяся от остальных процессоров не более чем на K дополнительных моделей.

Ш а г 3. Генерируем $P_\kappa(m)/2$ последовательных структур, начиная с той, которая отвечает десятичному числу $(\kappa-1)P_\kappa(m)/2$ и заканчивая – отвечающей десятичному числу $(\kappa-1)P_\kappa(m)/2 + P_\kappa(m)/2 = \kappa P_\kappa(m)/2$.

Ш а г 4. Находим структуру, инвертированную к той, которая отвечает десятичному числу $\kappa P_\kappa(m)/2$ (все единицы в двоичном представлении числа $\kappa P_\kappa(m)/2$ заменяем нулями и наоборот).

Ш а г 5. Генерируем $P_\kappa(m)/2$ последовательных структур, начиная с полученной на предыдущем шаге.

Оценка эффективности распараллеливания рекуррентных вычислений

Найдем теоретическую оценку эффективности разработанной схемы распараллеливания комбинаторного алгоритма с рекуррентным оцениванием параметров.

Пусть имеем m аргументов для полного перебора 2^m моделей на 2^k процессорах кластерной системы. Тогда на каждый процессор приходится дополнительная нагрузка, обусловленная необходимостью оценивать параметры первой модели нерекуррентно, в виде не более чем m моделей для первой половины последовательности двоичных структурных векторов и не более чем m моделей для второй половины последовательности, которая получается инвертированием предыдущих (см. шаги 4 и 5 приведенного алгоритма).

Каждый процессор построит 2^{m-k} моделей рекуррентно и дополнительно не более чем $2m$ моделей, что составляет $2m/2^{m-k} = m/2^{m-k-1}$ часть нагрузки.

При заданном количестве аргументов и растущем количестве процессоров теоретическая эффективность распараллеливания будет уменьшаться. Так, для $m = 25$ аргументов (для меньшего количества аргументов нет смысла распараллеливать алгоритм) и $2^7 = 128$ процессоров будем иметь 0,0002 (или 0,02 процента) потерь, т.е. теоретическая эффективность будет состав-

лять 99,98 процентов. При большем количестве аргументов она будет еще выше.

Кроме того, дополнительные потери будет иметь первый процессор в виде не более чем 2^{k+1} дополнительных моделей (см. шаг 2 алгоритма). Это означает дополнительную нагрузку на него, что составляет $2^{k+1}/2^{m-k} = 2^{2k-m+1}$ часть. Для случая $m = 25$ аргументов и $2^7 = 128$ процессоров будем иметь,0,001 (или 0,1 процента) потерь, т.е. общая теоретическая эффективность будет составлять 99,88 процентов.

Результаты экспериментов по рекуррентно-параллельным вычислениям

С целью экспериментального определения эффективности разработанной схемы комбинаторного алгоритма на основе рекуррентно-параллельных вычислений проведен тестовый эксперимент по решению задачи структурно-параметрической идентификации с помощью комбинаторного алгоритма с рекуррентно-параллельными вычислениями (для количества аргументов, равного 25) по разбиению всех вычислений на пять потоков и последовательным их выполнением на одном процессоре.

Результат этого эксперимента приближен к теоретическому, поскольку в нем исключено межпроцессорное взаимодействие – а в алгоритме *COMBI*, как было описано, такое взаимодействие отсутствует.

Эксперимент выполнялся следующим образом: генерировалась матрица плана X размером 45×25 (45 точек для 25 аргументов) для системы условных уравнений $X\theta = y$. Вектор y формировался в виде линейной комбинации пяти аргументов: $y = x_{11} + x_{12} + x_{13} + x_{14} + x_{15}$. Выполнялась структурно-параметрическая идентификация с выбором лучшей модели по критерию регулярности [1], а также измерялось время моделирования при распараллеливании на разное количество потоков.

Результат, представленный на рис. 4 в виде диаграммы времени выполнения, демонстрирует эффективность использования разработанной схемы.

Эффективность распараллеливания E достигла 99,7 процентов, а равномерность нагрузки P равнялась 99,2 процентам, которые определялись по формулам:

$$E = \frac{T_1}{5 \times T_{5\max}} \times 100\%, \qquad (7)$$

$$P = (1 - \frac{T_{5\max} - T_{5\min}}{T_{5\max}}) \times 100\%, \qquad (8)$$

где T_1 – время выполнения алгоритма с одним потоком (т.е. без распараллеливания), $T_{5\max}$ – время выполнения алгоритма с распараллеливанием на пять потоков (определяется как максимальное среди пяти потоков время выполнения программы), $T_{5\min}$ – минимальное среди пяти потоков время выполнения программы.

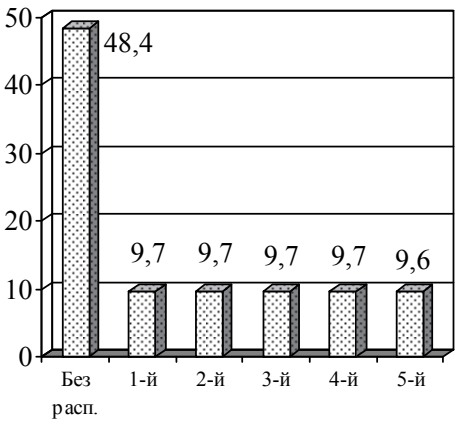

Рис. 4. Время (секунды) выполнения алгоритма *COMBI* на основе рекуррентно-параллельных вычислений

Исходя из теоретически двукратного увеличения времени моделирования при добавлении одного дополнительного аргумента для полного перебора и зная время выполнения алгоритма *COMBI* с рекуррентными вычислениями для определенного количества аргументов, можно оценить время моделирования на произвольном заданном количестве процессоров, что, естественно, целесообразно сделать до начала моделирования. Такие оценки для количества аргументов от тридцати до сорока приведены на рис. 5 и 6. Как видим, эффективность распараллеливания равна количеству процессоров (в нашем случае – стократная).

Заключение. Исследован процесс распараллеливания операций в комбинаторном алгоритме *COMBI* МГУА с рекуррентным вычислением параметров моделей. Разработана схема распараллеливания, базирующаяся на основе

алгоритма генерации стандартного двоичного счетчика, который отвечает последовательным десятичным числам.

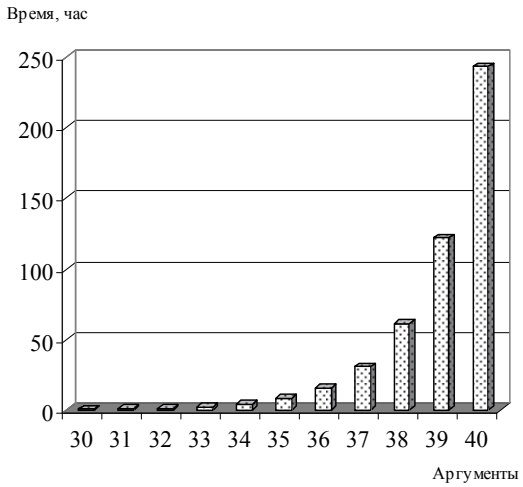

Рис. 5. Оценка времени выполнения комбинаторного алгоритма с рекуррентными вычислениями на одном процессоре

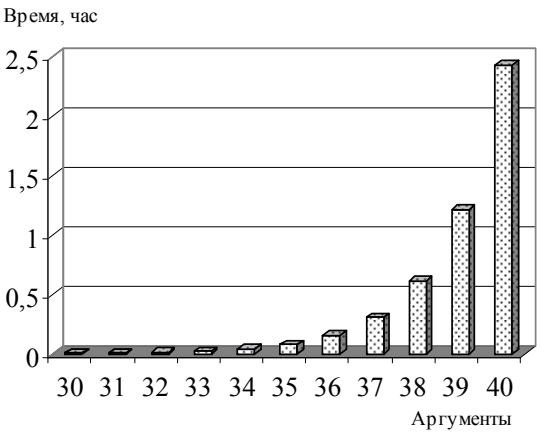

Рис. 6. Оценка времени выполнения комбинаторного алгоритма с рекуррентными вычислениями на ста процессорах

Особенность схемы состоит в том, что перед началом моделирования каждый процессор вычислительного кластера самостоятельно рассчитывает начальный и конечный структурный вектор для каждой сложности моделей, чем обеспечивается практически равномерная нагрузка на все элементы кластерной системы в целом и отпадает необходимость в межпроцессорном взаимодействии. Для задачи оценивания параметров по МНК использована рекуррентная модификация алгоритма Гаусса решения систем линейных уравнений.

С помощью тестового эксперимента показано, что использование предложенной схемы обеспечивает практически одинаковое суммарное количество моделей и оцениваемых параметров, которые приходятся на каждый процесс. Эффективность применения схемы при количестве процессов, равном пяти, и количестве аргументов, равном 25, составляет 99,7 процентов, а равномерность нагрузки достигает 99,2 процента.

С увеличением количества аргументов эффективность комбинаторного алгоритма с рекуррентно-параллельными вычислениями по отношению к параллельному алгоритму с нерекуррентным оцениванием параметров растет практически линейно.

1. *Ивахненко А.Г.*, *Степашко В.С.* Помехоустойчивость моделирования. – Киев: Наук. думка, 1985. – 216 с.
2. *Себер Дж.* Линейный регрессионный анализ. – М.: Мир, 1980. – 456 с.
3. *Степашко В.С.*, *Ефименко С.Н.* О последовательном оценивании параметров регрессионных моделей // Кибернетика и системный анализ. – 2005. – № 4. – С. 184–187.
4. *Stepashko V.*, *Yefimenko S.* Parallel algorithms for solving combinatorial macromodelling problems // Przegląd Elektrotechniczny (Electrical Review). – 2009. – **85**, N 4. – P. 98–99.
5. *Yefimenko S.* Comparative Effectiveness of Parallel and Recurrent Calculations in Combinatorial Algorithms of Inductive Modelling // Proc. of the 4th Int. Conf. on Inductive Modelling ICIM'2013. – Kyiv, 2013. – P. 231–234.
6. *Єфименко С.М.*, *Степашко В.С.* Рекурентний алгоритм методу Гаусса для розв'язання систем лінійних рівнянь у задачі оцінювання параметрів регресійних моделей // Відбір і обробка інформації: Міжв. зб. наук. пр. – 2012. – № 36 (112). – С. 48–55.

Поступила 30.10.2014
Тел. для справок: 044 526-3028, 097 105-8796 (Киев)
E-mail: *syefim@uk.net*