МАСШТАБИРУЕМЫЙ ПАРАЛЛЕЛЬНЫЙ АЛГОРИТМ ПОСТРОЕНИЯ ПСЕВДОПРОЕКЦИЙ В ЗАДАЧАХ СИЛЬНОЙ ОТДЕЛИМОСТИ

А.В. Ершова, И.М. Соколинская

1. Введение. Задача сильной отделимости – это задача нахождения слоя максимальной толщины, разделяющего два выпуклых многогранника. Эта задача имеет большое значение теоретического и прикладного характера в распознавании образов, включающем задачи дискриминации, классификации и др. Задача сильной отделимости может быть решена в ходе итерационного процесса, использующего операцию проектирования. Однако на практике применение этого метода существенно ограничивается тем, что далеко не всегда удается построить конструктивную формулу для вычисления проекции точки на выпуклое множество. Поэтому целесообразно заменить операцию проектирования последовательностью фейеровских отображений [1]. Указанный метод был описан в работе [2].

Алгоритмы разделения многогранников на базе фейеровских отображений обладают тем преимуществом по сравнению с другими известными методами, что они применимы к нестационарным задачам, то есть к задачам, в которых исходные данные могут меняться в процессе решения задачи. Примерами таких нестационарных задач являются, например, задача о портфеле ценных бумаг и задача о спам-фильтре. При решении *задачи о портфеле ценных бумаг* [3] является важным вопрос о выборе ценной бумаги из их многообразия при формировании портфеля. Проблему выбора можно решить, условно разделив все ценные бумаги на две части: прибыльные и убыточные. Бумага, которая будет дорожать, считается прибыльной, а бумага, которая будет дешеветь, – убыточной. Каждая ценная бумага представляется в виде многомерной точки, координаты которой – это параметрическое описание конкретной бумаги [4]. Эксперт брокерской компании определяет набор параметров, исходя из которых, он может предсказать, будет дорожать или дешеветь бумага. Такими параметрами могут быть положение бумаги в тренде, средний оборот торгов, спрэд между ценами спроса и предложения и др. Количество параметров определяет размерность пространства решаемой задачи. Эксперт формирует два множества точек: множество прибыльных бумаг и множество убыточных бумаг. Для каждого множества строится выпуклая оболочка в виде многогранника. Так мы приходим к задаче сильной отделимости. Построив слой наибольшей толщины, разделяющий два многогранника, мы получаем инструмент, позволяющий автоматически разделять ценные бумаги на прибыльные и убыточные.

В *задаче о спам-фильтре* [5] мы должны для каждого электронного письма определить, является данное письмо «спамом», или нет. Для этого также вводится система метрик. Пользователь разделяет свои письма и отмечает те, которые считает спамом [6]. На основании этой информации строятся выпуклые оболочки в виде систем линейных неравенств. Первая система задает множество точек-писем, определяемых как спам, вторая система – множество точек-писем, определяемых как не спам. Построив слой наибольшей толщины, разделяющий два многогранника, мы получаем инструмент, позволяющий автоматически разделять письма на «хорошие» и «плохие». Когда приходит новое письмо, считываются характеристики этого письма, получается точка в рассматриваемом пространстве. Если данная точка попадает в «плохое» полупространство, мы делаем предположение, что это спам; если в «хорошее» – не спам. Если точка попадает внутрь слоя, письмо доставляется пользователю с пометкой «возможно, спам».

При решении практических задач экономико-математического моделирования часто встречаются задачи с большим количеством переменных и ограничений. Размер типичной средней задачи может составлять 20 000 переменных и 5 000 ограничений [7]. В отдельных случаях количество переменных может превышать 100 000, а количество ограничений – 20 000. Подобные задачи при решении требуют значительных вычислительных мощностей. В связи с этим актуальной является задача разработки параллельного алгоритма разделения выпуклых многогранников с помощью фейеровских отображений, допускающего эффективную реализацию на многопроцессорных системах с массовым параллелизмом.

В данной работе предлагается новый масштабируемый параллельный алгоритм для решения задачи сильной отделимости, основанный на использовании фейеровских отображений. Рассматриваются теоретические и реализационные аспекты параллельного алгоритма. Приводятся результаты вычислительных экспериментов на многопроцессорной системе с кластерной архитектурой, подтверждающие эффективность предложенного подхода.

2. Математическая модель. Пусть даны два выпуклых непересекающихся многогранника $M \subset \mathbb{R}^n$ и $N \subset \mathbb{R}^n$, заданные системами линейных неравенств:

$$M = \{x | AX \leqslant b\} \neq \varnothing,$$
$$N = \{x | BX \leqslant d\} \neq \varnothing, \quad (1)$$
$$M \cap N = \varnothing.$$

Задача сильной отделимости – это задача нахождения слоя наибольшей толщины, разделяющего *M* и *N*. Сильная отделимость, по существу, эквивалентна задаче отыскания расстояния между *M* и *N* в смысле метрики

$$\rho(M, N)=min\{\|x-y\| \mid x\in M, y\in N\}. \qquad (2)$$

Если $\bar{x}\in M$ и $\bar{y}\in N$ являются arg-точками задачи (2), то есть $\rho(M, N)=\|x-y\|$, то слоем наибольшей толщины, разделяющим множества *M* и *N*, является $P:=\{x \mid x\in P_1\cap P_2\}$, где P_1 и P_2 – полупространства, задаваемые линейными неравенствами

$$(x-\bar{x}, \bar{x}-\bar{y})\leqslant 0 \text{ и } (y-\bar{y}, \bar{x}-\bar{y})\geqslant 0.$$

Задача сильной отделимости может быть решена с помощью известного метода *последовательного проектирования* [2].

Если множества *M* и *N* достаточно просты в смысле простоты реализации операции проектирования точек на них, то алгоритм на базе последовательного проектирования может быть использован на практике. Но если *M* и *N* – произвольные многогранники, то такой алгоритм не может быть признан эффективным, так как не известен универсальный конструктивный метод построения проекции точки на многогранник. Ситуацию можно исправить, если вместо операции проектирования использовать фейеровские отображения.

Дадим определение фейеровского отображения. Пусть $\varphi\in\{\mathbb{R}^n\to\mathbb{R}^n\}$. Отображение φ называется *M-фейеровским*, если выполняются следующие два условия:

$$\varphi(y)=y, \ \forall y\in M;$$
$$\|\varphi(x)-y\|<\|x-y\|, \ \forall y\in M, \ \forall x\notin M.$$

Сконструируем *M*-фейеровское отображение, следуя работе [8]. Представим систему линейных неравенств, задающих многогранник *M*, в следующем виде:

$$Ax\leqslant b: l_j(x)=(a_j, x)-b_j\leqslant 0, j=1,\ldots, m, \quad (3)$$

где $a_j\neq 0$ для любого j. Определим $l_j^{plus}(x)$ следующим образом:

$$l_j^{plus}(x)=max\{l_j(x), 0\}, j=1,\ldots, m. \qquad (4)$$

Тогда отображение вида

$$\varphi(x)=x-\sum_{j=1}^{m}\alpha_j\lambda_j\frac{l_j^{plus}(x)}{\|a_j\|^2}a_j \qquad (5)$$

будет *M*-фейеровским для любой системы положительных коэффициентов $\{\alpha_j>0\}$, $j=1,\ldots, m$, таких, что $\sum_{j=1}^{m}\alpha_j=1$ и коэффициентов релаксации $0<\lambda_j<2$. Аналогичным образом сконструируем *N*-фейеровское отображение ψ. Используя отображения φ и ψ, мы можем построить следующий алгоритм **F**, решающий задачу сильной отделимости с использованием фейеровских отображений.

Алгоритм F. Пусть задано произвольное начальное приближение $z_0\in\mathbb{R}^n$. Зафиксируем положительное вещественное число ε. Алгоритм состоит из следующих шагов:

Шаг 0. $k:=0$.

Шаг 1. $x_{k+1}:=\lim_{u\to\infty}\varphi^u(z_k)$.

Шаг 2. $y_{k+1}:=\lim_{u\to\infty}\psi^u(z_k)$.

Шаг 3. $z_{k+1}:=\frac{x_{k+1}+y_{k+1}}{2}$.

Шаг 4. $k:=k+1$.

Шаг 5. Если $min\{\|x_{k+1}-x_k\|, \|y_{k+1}-y_k\|\}\geqslant\varepsilon$, перейти на шаг 1.

Шаг 6. Стоп.

Алгоритм **F** был исследован в работе [9]. Проведенные вычислительные эксперименты на модельных и случайных задачах подтвердили его эффективность. Однако, для больших размерностей работа алгоритма **F** требовала значительного времени. Например, при размерности задачи *n* = 512 время счета на одном процессорном ядре составило 16 часов, а при размерности задачи *n* = 1024 – 608 часов (более 25 дней). В связи с этим возникла необходимость разработки параллельной версии этого алгоритма для многопроцессорных систем с массовым параллелизмом. Очевидно, что в алгоритме **F** ресурсоемкими являются шаги 1 и 2. На каждом из этих шагов реализуется *последовательный* фейеровский процесс, в результате которого мы получаем *псевдопроекцию* точки на многогранник. Наши исследования показали, что подобные фейеровские процессы не

допускают эффективного распараллеливания на большом количестве процессорных узлов (предел масштабируемости в экспериментах не превышал 8-16 узлов). В следующем разделе мы предложим новый масштабируемый алгоритм построения псевдопроекции точки на многогранник с использованием фейеровских отображений.

3. Параллельный алгоритм построения псевдопроекции. В основе параллельного алгоритма построения псевдопроекции на многогранник лежит метод разбиения вектора на подвекторы. Для каждого подвектора организуется независимый фейеровский процесс. Через каждые *K* шагов результаты, полученные на подвекторах, соединяются в один вектор, для которого считается *невязка*. Если значение невязки меньше заданного положительного числа ε, то полученный вектор принимается в качестве псевдопроекции. В противном случае вычисления продолжаются.

Дадим формальное описание алгоритма построения псевдопроекции на выпуклый многогранник с помощью фейеровских отображений, допускающего эффективное распараллеливание на большом количестве процессорных узлов.

Введем следующие обозначения. Для произвольного линейного подпространства $\boldsymbol{P} \subset \mathbb{R}^n$ через $\pi_{\boldsymbol{P}}(x)$ будем обозначать ортогональную проекцию $x \in \mathbb{R}^n$ на линейное подпространство $\boldsymbol{P}$. Везде далее линейное подпространство мы будем называть просто подпространством. Через $\rho(\boldsymbol{P}, x) := \min\limits_{p \in \boldsymbol{P}} \| p - x \|$ будем обозначать расстояние от точки x до подпространства $\boldsymbol{P}$. Пусть линейное многообразие $\boldsymbol{L}$ получается из $\boldsymbol{P}$ сдвигом на некоторый вектор z: $\boldsymbol{L} = \boldsymbol{P} + z$. Через $\pi_{\boldsymbol{L}}(x)$ будем обозначать ортогональную проекцию $x \in \mathbb{R}^n$ на линейное многообразие $\boldsymbol{L}$: $\pi_{\boldsymbol{L}}(x) = \pi_{\boldsymbol{P}}(x) + z$. Теперь мы готовы к формальному описанию алгоритма.

Алгоритм S. Пусть задано однозначное *M*-фейеровское отображение $\varphi \in \{\mathbb{R}^n \to \mathbb{R}^n\}$, *M* – выпукло и замкнуто. Зададим разбиение пространства $\mathbb{R}^n$ в прямую сумму ортогональных подпространств: $\mathbb{R}^n = \boldsymbol{P_1} + \ldots + \boldsymbol{P_r}$, $\boldsymbol{P_i} \perp \boldsymbol{P_j}$ при $i \neq j$. Для каждого подпространства $\boldsymbol{P_i} (i = 1, \ldots, r)$ построим линейное многообразие $\boldsymbol{L_i}$ следующим образом. Пусть $\bar{x}_i \in Arg \min\limits_{x \in M} \rho(\boldsymbol{P_i}, x)$. Разложим $\bar{x}_i$ в сумму ортогональных векторов $\bar{x}_i = \pi_{\boldsymbol{P_1}(\bar{x}_i)} + \ldots + \pi_{\boldsymbol{P_r}(\bar{x}_i)}$ из подпространств $\boldsymbol{P_1}, \ldots, \boldsymbol{P_r}$. Обозначим $z_i = \pi_{\boldsymbol{P_i}}(\bar{x}_i) \in \boldsymbol{P_i}$. Построим линейное многообразие $\boldsymbol{L_i}$ путем сдвига подпространства $\boldsymbol{P_i}$ на вектор z_i:

$$\boldsymbol{L_i} = \boldsymbol{P_i} + z_i . \quad (6)$$

Для каждого $i \in \{1, \ldots, r\}$ определим отображение

$$\varphi_i(x) = \pi_{\boldsymbol{L_i}}(\varphi(\pi_{\boldsymbol{L_i}}(x))) . \quad (7)$$

Зафиксируем некоторое натуральное число *s* и положительное вещественное число ε. Зададим произвольное начальное приближение $x_0 \in \mathbb{R}^n$. Алгоритм состоит из следующих шагов:

Шаг 0. $k := 0$.

Шаг 1. $x_{k+1} := \sum\limits_{i=1}^{r} (\varphi_i^s(x_k) - \bar{z}^i)$.

Шаг 2. $d := \sum\limits_{j=1}^{m} l_j^{plus}(x_{k+1})$.

Шаг 3. $k := k + 1$.

Шаг 4. Если $d \geqslant \varepsilon$ и $\|x_{k+1} - x_k\| \neq 0$, перейти на шаг 1.

Шаг 5. Стоп.

На шаге 2 вычисляется невязка *d*, которая определяет степень близости точки x_{k+1} к многограннику *M*.

Следующая теорема показывает, что каждое отображение $\varphi_i \in \{\boldsymbol{L_i} \to \boldsymbol{L_i}\}$, строящееся в алгоритме **S**, является фейеровским относительно множества $\boldsymbol{L_i} \cap M$.

Теорема 1. Пусть задано замкнутое множество $M \subset \mathbb{R}^n$ и однозначное *M*-фейеровское отображение $\varphi \in \{\mathbb{R}^n \to \mathbb{R}^n\}$. Пусть $\boldsymbol{P}$ – некоторое собственное линейное подпространство в пространстве $\mathbb{R}^n$, $\boldsymbol{T} = \boldsymbol{P}^{\cdot\perp}$ – ортогональное дополнение к пространству $\boldsymbol{P}$. Пусть $\bar{x} \in Arg \min\limits_{x \in M} \rho(\boldsymbol{P}, x)$. Представим $\bar{x}$ в виде суммы ортогональных векторов из подпространств $\boldsymbol{P}$ и $\boldsymbol{T}$: $\bar{x} = \pi_{\boldsymbol{P}}(\bar{x}) + \pi_{\boldsymbol{T}}(\bar{x})$. Обозначим $z = \pi_{\boldsymbol{T}}(\bar{x}) \in \boldsymbol{T}$.

Построим линейное многообразие $\boldsymbol{L}$ путем сдвига подпрстранства $\boldsymbol{P}$ на вектор z: $\boldsymbol{L}=\boldsymbol{P}+z$. Определим отображение $\varphi_{\boldsymbol{L}}\in\{\boldsymbol{L}\to\boldsymbol{L}\}$ следующим образом:

$$\varphi_{\boldsymbol{L}}(x)=\pi_{\boldsymbol{L}}(\varphi(\pi_{\boldsymbol{L}}(x))). \qquad (8)$$

Положим

$$M_{\boldsymbol{L}}=\boldsymbol{L}\cap M. \quad (9)$$

Тогда отображение $\varphi_{\boldsymbol{L}}$ является $M_{\boldsymbol{L}}$-фейеровским.

Для того, чтобы алгоритм $\mathbf{S}$ был корректным, необходимо и достаточно, чтобы последовательность точек $\{x_k\}$, порождаемых алгоритмом $\mathbf{S}$ на шаге 1, сходилась.

Теорема 2. Пусть $\{x_k\}$ – последовательность точек, порождаемых алгоритмом $\mathbf{S}$ на шаге 1:

$$x_{k+1}:=\sum_{i=1}^{r}\varphi_i^s(x_k). \qquad (10)$$

Тогда

$$\{x_k\}_{k=0}^{+\infty}\to\bar{x}\in\mathbb{R}^n.$$

Таким образом, на базе одного фейеровского отображения мы сконструировали другое, обладающее большим ресурсом параллелизма. Действительно, значения $\varphi_i^s(x)$ могут вычисляться независимо друг от друга для различных $i=1,\ldots,r$. При этом мы получаем две степени свободы для регулировки баланса загрузки процессорных узлов. Во-первых, увеличивая *s*, мы можем повышать вычислительную нагрузку на процессорный узел между двумя соседними итерациями, вычисляющими $\varphi_i(x)$. Во-вторых, мы можем произвольным образом перераспределять координаты вектора x между процессорами.

При реализации параллельного алгоритма $\mathbf{S}$ применение фейеровского отображения *s* раз к каждому подвектору оформлялось в виде независимого процесса. Далее происходил обмен вычисленными $\varphi_i^s(x)$ между процессорами. Указанный подход позволяет без какой-либо модификации выполнять параллельную программу на различном количестве процессоров, не превосходящем количество подвекторов. Для организации обменов данными между процессами была использована система параллельного программирования MPI, основанная на передаче сообщений [10]. Исходные тексты программ доступны в Интернет по адресу http://life.susu.ru/discr/.

4. Вычислительные эксперименты. Для проведения вычислительных экспериментов мы использовали сконструированную нами *модельную задачу Model-n.* Для такого типа задач можно легко аналитически вычислить точное значение толщины максимального разделяющего слоя. Поэтому они хорошо подходят для проверки корректности алгоритма и изучения его масштабируемости. Кроме того, модельные задачи дают хорошую возможность для подбора оптимальных значений параметров алгоритма. Задача *Model-n* имеет следующий вид (*n* задает размерность задачи):

Многогранник M	Многогранник N
$x_1-2x_2\le 0$	$x_1-2x_2\le 20000$
$x_1-2x_3\le 0$	$x_1-2x_3\le 20000$
…	…
$x_1-2x_n\le 0$	$x_1-2x_n\le 20000$
$x_1+2x_2\le 20000$	$x_1+2x_2\le 40000$
$x_1+2x_3\le 20000$	$x_1+2x_3\le 40000$
…	…
$x_1+2x_n\le 20000$	$x_1+2x_n\le 40000$
$-x_1\le 0$	$-x_1\le -20000$
$-x_2\le 0$	$-x_2\le 0$
…	…
$-x_n\le 0$	$-x_n\le 0$

Для всех задач *Model-n* толщина максимального разделяющего слоя равна 10 000. На **Рис. 1** изображены многогранники *M* и *N*, задаваемые задачей *Model*-3 ($n=3$), и итерации последовательного алгоритма $\mathbf{F}$, приводящие к циклу неподвижности.

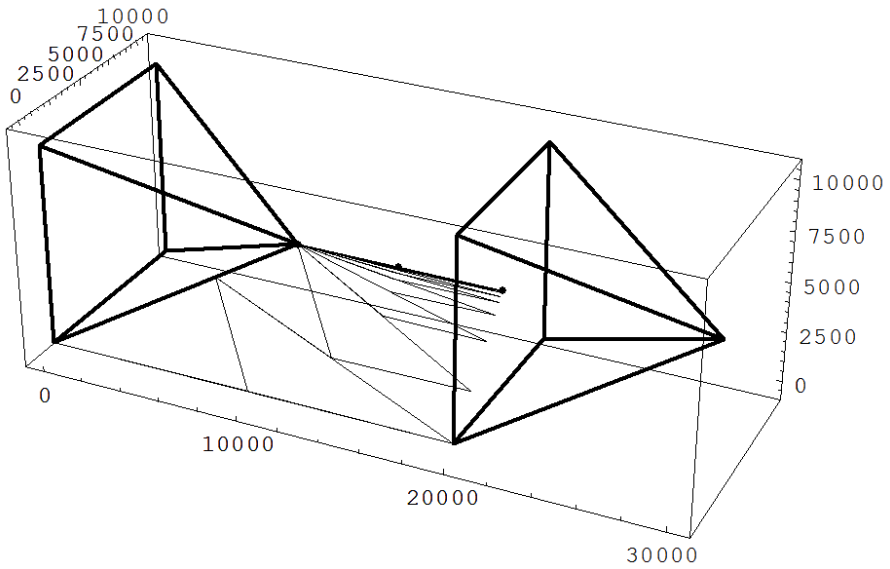

Рис. 1. Итерации последовательного алгоритма F для задачи Model-3.

Для проведения экспериментов был использован суперкомпьютер «СКИФ-Урал» (с кластерной архитектурой) с характеристиками, представленными в Таблице 1.

Таблица 1. Характеристики вычислительного кластера.

Число лезвий blade-системы / процессоров / ядер	166 / 332 / 1328
Тип процессора	Intel Xeon E5472 (4 ядра по 3.0 GHz)
Оперативная память	1.33 TB
Дисковая память	49.29 TB
Система хранения данных	Panasas ActiveStorage 5100 (20 TB)
Тип системной сети	InfiniBand (20Gbit/s, макс. задержка 2 мкс)
Тип управляющей (вспомогательной) сети	Gigabit Ethernet
Сервисная сеть	СКИФ ServNet
Пиковая производительность	16 TFlops
Производительность на тесте LINPACK	12.2 TFlops
Операционная система	SUSE Linux Enterprise Server 10
	Windows HPC Server 2008
Система бесперебойного электропитания	APC Symmetra 160 kVA

В первой серии вычислительных экспериментов исследовалась зависимость времени решения задачи от параметра s. Эксперименты проводились при фиксированной размерности задачи $n = 512$. Значения s варьировались от 150 до 1450. При этом задача вычислялась на разном количестве процессоров: $p = 64$, $p = 128$ и $p = 256$. Результаты экспериментов показывают (Рис. 2), что при увеличении значения s время решения задачи уменьшается на несколько порядков. Это происходит вследствие того, что с ростом s уменьшается количество межпроцессорных обменов. Однако при достижении определенных величин дальнейшего существенного ускорения счета не происходит. При количестве процессорных ядер p = 64 этот порог достигается при $s = 550$, для p = 128 пороговым значением будет $s = 800$, для p = 256 пороговым значением будет $s = 950$. Таким образом, при большом количестве задействованных процессорных узлов увеличение значения параметра s приводит к более значительному уменьшению времени счета.

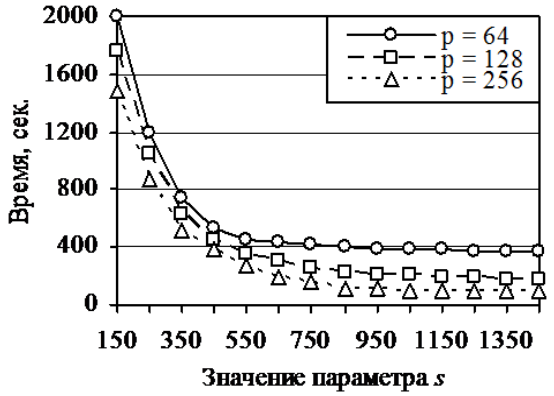

Рис. 2. Зависимость времени решения задачи от параметра *s*.

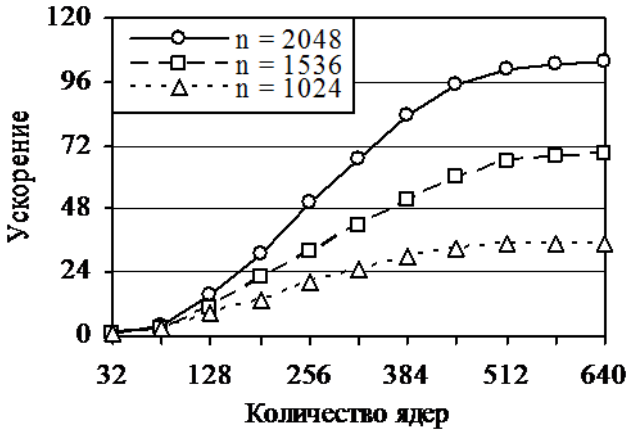

Рис. 3. Ускорение для задачи *Model-n.*

Во второй серии экспериментов (Рис. 3) исследовалось ускорение параллельного алгоритма. Ускорение вычислялось по формуле $\alpha = t_p / t_{32}$, где t_{32} – время, затраченное на решение задачи *Model*-n на 32 процессорных ядрах, t_p – время, затраченное на решение этой же задачи на p процессорных ядрах. Эксперименты проводились для трех размерностей: $n = 1024$, $n = 1536$ и $n = 2048$, при фиксированном $s = 10\,000$. Для каждой размерности варьировалось количество процессорных ядер, используемых для решения задачи. Во всех трех случаях мы наблюдали ускорение, близкое к линейному, вплоть до 512 процессорных ядер. Далее рост ускорения замедлялся. Однако, для больших значений размерности задачи n кривая ускорения приобретала горизонтально-асимптотический характер существенно позднее.

В третьей серии вычислительных экспериментов варьировался процент обменов между процессорными ядрами. Рассматривалась параллельная реализация итерационного алгоритма нахождения фейеровского отображения точки на многогранник. Нами был введен параметр ω, определяемый как интенсивность обменов между процессорными ядрами, измеряемый в процентах.

В первой части экспериментов исследовалась зависимость отклонения Δ от интенсивности ω. Отклонение Δ определялось как расстояние между полученным при выполнении параллельного алгоритма решением и решением задачи в результате работы последовательного алгоритма. Эксперименты проводились для трех размерностей: $n = 256$, $n = 512$ и $n = 1024$ при фиксированном количестве ядер $p = 32$. Интенсивность ω варьировалась от 0 до 100. При 100% обменов параллельный алгоритм производит обмен данными между процессорными ядрами на каждой итерации. При 0% – обменов данными не происходит до конца работы алгоритма. Результаты эксперимента представлены на **Рис. 4**.

Графики показывают, что при увеличении процента обменов отклонение Δ от точного решения уменьшается для всех трех размерностей и становится равным нулю при 100% обменов. Кроме того, чем больше размерность решаемой задачи, тем меньше величина отклонения Δ.

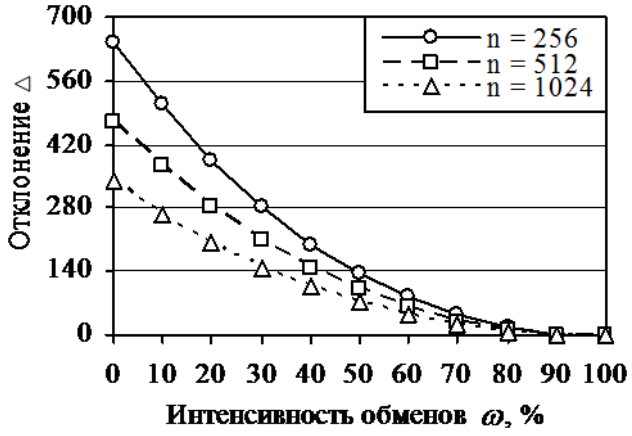

Рис. 4. Зависимость отклонения Δ от интенсивности ω.

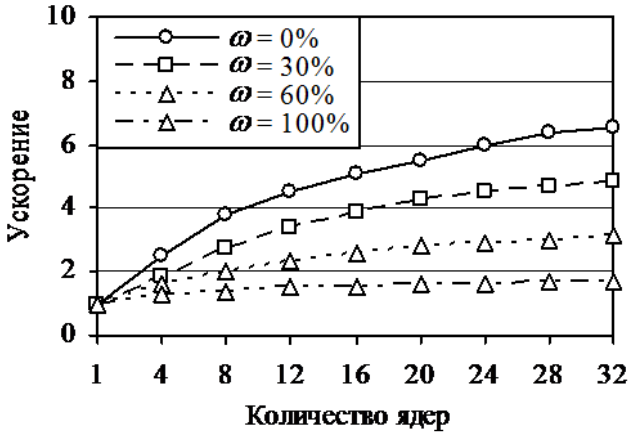

Рис. 5. Кривые ускорения при разной интенсивности обменов ω.

Во второй части экспериментов исследовалось ускорение параллельного алгоритма при различной

интенсивности ω . Задача решалась для четырех вариантов алгоритма с интенсивность ω равной 0%, 30%, 60% и 100%, при фиксированной размерности задачи $n = 512$. Для каждого варианта алгоритма варьировалось количество процессорных ядер, используемых для решения задачи. Результаты экспериментов приведены на Рис. 5.

Графики показывают, что с уменьшением процента обменов между процессорными ядрами ускорение при работе параллельного алгоритма возрастает, а наилучшее ускорение достигается при работе варианта алгоритма, вообще не использующего обмены.

5. Заключение. В данной работе были рассмотрены итерационные алгоритмы **F** и **S** для решения задачи разделения двух выпуклых непересекающихся многогранников слоем наибольшей толщины. Алгоритм **F** базируется на использовании фейеровских отображений в качестве аналога операции проектирования. Предложен новый параллельный алгоритм **S**, который может быть использован без каких-либо изменений как на многопроцессорных системах с общей памятью, так и на системах с массовым параллелизмом, включая кластерные системы. Проведены вычислительные эксперименты на высокопроизводительном кластере по исследованию масштабируемости алгоритма **S** на модельных задачах *Model-n* и по исследованию масштабируемости параллельной версии итерационного алгоритма нахождения фейеровского отображения точки на многогранник с различными вариантами процента обменов между процессорными ядрами. Проведенные вычислительные эксперименты подтверждают эффективность предложенных подходов. Исходные тексты программ свободно доступны в Интернет по адресу: http://life.susu.ru/discr/.

Описанный параллельный алгоритм может быть использован для создания программы-робота, которая будет в автоматическом режиме совершать биржевые операции, обеспечивая стабильную доходность. В настоящее время на мировых биржах уже трудятся сотни программ-роботов. Это явление получило название «роботрейдинг». Такой алгоритм позволит создать программу-робота нового поколения, которая может быстро реагировать на изменяющиеся условия рынка ценных бумаг и валют, обеспечивая владельцу преимущество в конкурентной борьбе [11].

В рамках дальнейших исследований мы предполагаем реализовать новый метод решения задачи разделения двух выпуклых непересекающихся многогранников на основе гибридной технологии использования фейеровских отображений и операции проектирования многомерной точки на выпуклый многогранник. Кроме этого, мы планируем тестирование разработанного программного комплекса на реальных задачах большой размерности.

Работа поддержана грантом РФФИ № 09-01-00546а.

ЛИТЕРАТУРА:

1. И.И. Еремин, В.Д. Мазуров Нестационарные процессы математического программирования. М.: Наука, 1979.
2. И.И. Еремин Фейеровские методы сильной отделимости выпуклых полиэдральных множеств // Известия вузов. Сер. математика, 2006. № 12. с.33-43.
3. H. Markowitz Portfolio selection // Journal of Finance. 1952. Vol.7, № 1.
4. В.А. Боровкова Рынок ценных бумаг. СПб: Питер, 2005.
5. L. Zhang, J. Zhu, T. Yao An Evaluation of Statistical Spam Filtering Techniques // Transactions on Asian Language Information Processing, 2004. Vol. 3, No. 4.
6. A. Lampert, R. Dale, C. Paris Segmenting Email Message Text into Zones // Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: ACL and AFNLP, 2009. p.919-928.
7. J.J.H. Forrest, J.A. Tomlin Implementing the simplex method for the optimization subroutine library // IBM Systems Journal. 1992. Vol. 31, No. 1. p.11-25.
8. И.И. Еремин Теория линейной оптимизации. Екатеринбург: Изд-во “Екатеринбург”, 1999.
9. А.В. Ершова Алгоритм разделения двух выпуклых непересекающихся многогранников с использованием фейеровских отображений // Системы управления и информационные технологии, 2009. №1(35). с.53-56.
10. J.J. Dongarra, S.W. Otto, M. Snir, D. Walker A message passing standard for MPP and workstations // Communications of the ACM. 1996. Vol. 39, No. 7. p.84-90.
11. С.И. Майоров О современных тенденциях развития торговых технологий // Информационно-аналитический журнал «Биржевое обозрение», 2009. №10(70). с.14-17.