

# ПАРАЛЛЕЛЬНАЯ РЕАЛИЗАЦИЯ СЛЕДЯЩЕГО АЛГОРИТМА ДЛЯ РЕШЕНИЯ НЕСТАЦИОНАРНЫХ ЗАДАЧ ЛИНЕЙНОГО ПРОГРАММИРОВАНИЯ<sup>1</sup>

*И.М. Соколинская, Л.Б. Соколинский*

В статье описывается параллельный алгоритм решения нестационарных задач линейного программирования большой размерности, ориентированный на кластерные вычислительные системы. В основе алгоритма, получившего название «следающий», лежат фейеровские отображения. Алгоритм отслеживает изменения исходных данных и вносит корректировки в вычислительный процесс. При этом задача разбивается на большое количество подзадач, которые могут решаться независимо без обменов данными. Приводятся диаграммы деятельности UML, описывающие реализацию следающего алгоритма.

*Ключевые слова:* нестационарная задача линейного программирования, фейеровские отображения, следающий алгоритм, диаграммы деятельности UML, массовый параллелизм, кластерные вычислительные системы.

## ОБРАЗЕЦ ЦИТИРОВАНИЯ

Соколинская И.М., Соколинский Л.Б. Параллельная реализация следающего алгоритма для решения нестационарных задач линейного программирования // Вестник ЮУрГУ. Серия: Вычислительная математика и информатика. 2016. Т. 5, № 2. С. 15–29. DOI: 10.14529/cmse160202.

## Введение

Основы современной теории нестационарных процессов математического программирования были заложены в классической монографии [1], где было предложено использовать для решения нестационарных задач линейного программирования итерационные процессы фейеровского типа. Указанный подход обобщает релаксационный метод Моцкина-Агмона [2, 3]. Это направление исследований получило продолжение в дальнейших работах И.И. Еремина, В.В. Васина, Л.Д. Попова, Е.А. Бердниковой, С.В. Пацко и других ученых [4].

Нестационарные задачи линейного программирования большой размерности с быстро меняющимися входными данными достаточно часто встречаются в практике современного экономико-математического моделирования. Одним из примеров таких задач является задача управления портфелем ценных бумаг с использованием методов алгоритмической торговли [5, 6]. В подобных задачах количество переменных и неравенств в системе ограничений может составлять десятки и даже сотни тысяч, а период изменения исходных данных находится в пределах сотых долей секунды. На нестационарность в таких задачах может накладываться плохая формализуемость части ограничений. В работе авторов [7] был описан параллельный алгоритм для решения задач линейного программирования с плохо формализуемыми ограничениями. Суть предложенного подхода заключается в синтезе методов линейного программирования и дискрими-

<sup>1</sup> Статья рекомендована к публикации Программным комитетом Международной научной конференции "Параллельные вычислительные технологии (ПаВТ) 2016" (<http://agora.guru.ru/pavt>).

нантного анализа. Для выполнения эффективного дискриминантного анализа необходимы два набора образцов  $M$  и  $N$ , первый из которых удовлетворяет неформализованным ограничениям, а второй – нет. Для получения качественных наборов образцов могут применяться методы интеллектуального анализа [8] данных и анализа временных рядов [9].

Для преодоления проблемы нестационарности входных данных в работах [10, 11] был предложен «следящий» алгоритм решения задачи линейного программирования с использованием фейеровских отображений, ориентированный на кластерные вычислительные системы с многоядерными ускорителями. В данной работе дается полная параллельная реализация следящего алгоритма с использованием диаграммы деятельности UML. Статья организована следующим образом. В разделе 1 приводится формальная постановка задачи линейного программирования, даются определения фейеровского процесса и операции псевдопроектирования на многогранник. В разделе 2 приводится математическое описание следящей области. В разделе 3 приводятся формулы для построения пересечения многогранника, задаваемого системой ограничений, с произвольной ячейкой следящей области. В разделе 4 дается полное описание параллельной реализации следящего алгоритма с помощью диаграмм деятельности UML. В заключении суммируются полученные результаты и определяются направления дальнейших исследований.

## 1. Постановка задачи

Пусть задана задача линейного программирования

$$\max \{ \langle c, x \rangle \mid Ax \leq b, x \geq 0 \}. \quad (1)$$

Определим фейеровское отображение  $\varphi : \mathbb{R}^n \rightarrow \mathbb{R}^n$  следующим образом:

$$\varphi(x) = x - \sum_{i=1}^m \alpha_i \lambda_i \frac{\max \{ \langle a_i, x \rangle - b_i, 0 \}}{\|a_i\|^2} \cdot a_i. \quad (2)$$

Пусть  $M$  – многогранник, задаваемый ограничениями задачи линейного программирования (1). Такой многогранник всегда является выпуклым. Известно [12], что  $\varphi$  будет однозначным непрерывным  $M$ -фейеровским отображением для любой системы положительных коэффициентов  $\{ \alpha_i > 0 \}$ ,  $i = 1, \dots, m$ , таких, что  $\sum_{i=1}^m \alpha_i = 1$ , и коэффициентов релаксации  $0 < \lambda_i < 2$ . Полагая в формуле (2)  $\lambda_i = \lambda$  и  $\alpha_i = 1/m$  ( $i = 1, \dots, m$ ), получаем формулу

$$\varphi(x) = x - \frac{\lambda}{m} \sum_{i=1}^m \frac{\max \{ \langle a_i, x \rangle - b_i, 0 \}}{\|a_i\|^2} \cdot a_i, \quad (3)$$

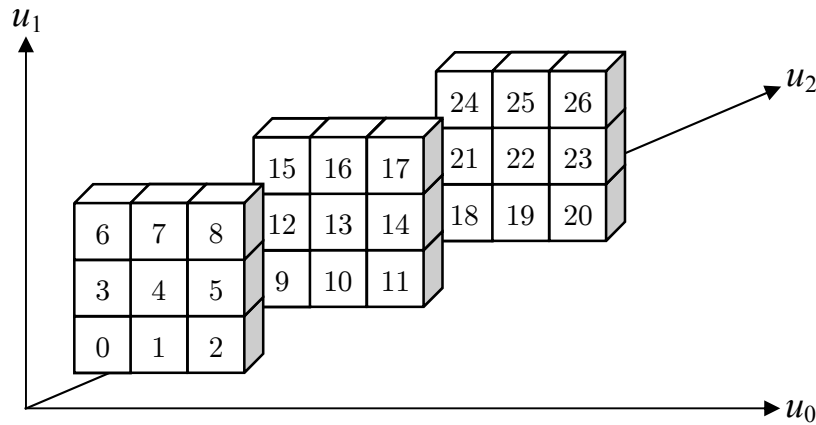
которая будет использоваться в следящем алгоритме.

Обозначим

$$\varphi^s(x) = \underbrace{\varphi \dots \varphi(x)}_s.$$

Под *фейеровским процессом*, порождаемым отображением  $\varphi$  при произвольном начальном приближении  $x_0 \in \mathbb{R}^n$ , будем понимать последовательность  $\{\varphi^s(x_0)\}_{s=0}^{+\infty}$ . Известно, что указанный фейеровский процесс сходится к точке, принадлежащей множеству  $M$ :

$$\{\varphi^s(x_0)\}_{s=0}^{+\infty} \rightarrow \bar{x} \in M. \quad (4)$$



**Рис. 1.** Линейная нумерация ячеек следящей области при  $n = 3$

Будем кратко обозначать это следующим образом:  $\lim_{s \rightarrow \infty} \varphi^s(x_0) = \bar{x}$ .

Под  $\varphi$ -проектированием (псевдопроектированием) точки  $x \in \mathbb{R}^n$  на многогранник  $M$  понимается отображение  $\pi_M^\varphi(x) = \lim_{s \rightarrow \infty} \varphi^s(x)$ .

## 2. Построение следящей области

Без ограничения общности мы можем считать, что все процессы происходят в положительной области координат. Пусть  $n$  – размерность пространства решений,  $K$  – количество ячеек в следящей области по одному измерению. Пусть  $P$  – количество МРІ-процессов, используемых для распараллеливания вычислений. Будем предполагать, что всегда выполняется равенство:

$$K^n = P, \quad (5)$$

то есть, количество ячеек следящей области равно количеству МРІ-процессов. Зададим в пространстве целочисленных координат  $u_0, \dots, u_{n-1}$  линейную нумерацию ячеек следящей области следующим образом. Пусть ячейка  $\alpha$  имеет целочисленные координаты  $(\alpha_0, \dots, \alpha_{n-1})$ . Тогда ее номер  $k_\alpha$  вычисляется по формуле:

$$k_\alpha = \sum_{i=0}^{n-1} \alpha_i K^i. \quad (6)$$

На рис. 1 приведен пример такой линейной нумерации при  $K = 3$ . Например, ячейка с номером 19 на рис. 1 имеет целочисленные координаты  $(1, 0, 2)$ . Действительно,  $19 = 1 \cdot 3^0 + 0 \cdot 3^1 + 2 \cdot 3^2$ .

Выразим целочисленные координаты ячейки  $\alpha$  через ее порядковый номер  $k_\alpha$ . Из (6) получаем

$$\alpha_0 = k_\alpha \bmod K; \quad (7)$$

$$\alpha_1 = \frac{k_\alpha - \alpha_0}{K} \bmod K; \quad (8)$$

$$\alpha_2 = \frac{k_\alpha - \alpha_0 - \alpha_1 K}{K^2} \bmod K; \quad (9)$$

.....

Таким образом, в общем виде имеем:

$$\alpha_i = \frac{k_\alpha - \sum_{j=0}^{i-1} \alpha_j K^j}{K^i} \bmod K. \quad (10)$$

Формула (10) содержит ресурсоемкую операцию возведения в степень. От нее можно избавиться следующим образом. По определению операции mod из (7) получаем

$$\alpha_0 = k_\alpha - (k_\alpha \div K) \cdot K. \quad (11)$$

Подставив в (8) вместо  $\alpha_0$  правую часть этого уравнения, получим

$$\begin{aligned} \alpha_1 &= \frac{k_\alpha - (k_\alpha - (k_\alpha \div K) \cdot K)}{K} \bmod K \\ &= \frac{(k_\alpha \div K) \cdot K}{K} \bmod K \\ &= (k_\alpha \div K) \bmod K. \end{aligned} \quad (12)$$

По определению операции mod отсюда следует

$$\alpha_1 = k_\alpha \div K - ((k_\alpha \div K) \div K) \cdot K. \quad (13)$$

Подставив в (9) вместо  $\alpha_0$  правую часть уравнения (11), а вместо  $\alpha_1$  – правую часть уравнения (13), получим

$$\begin{aligned} \alpha_2 &= \frac{k_\alpha - \alpha_0 - \alpha_1 K}{K^2} \bmod K \\ &= \frac{k_\alpha - (k_\alpha - (k_\alpha \div K) \cdot K) - (k_\alpha \div K - ((k_\alpha \div K) \div K) \cdot K) \cdot K}{K^2} \bmod K \\ &= ((k_\alpha \div K) \div K) \bmod K. \end{aligned} \quad (14)$$

Из (12) и (14) для  $i = 1, \dots, n-1$  по индукции получаем:

$$\alpha_i = \underbrace{(((k_\alpha \div K) \dots) \div K)}_i \bmod K. \quad (15)$$

Пусть  $g = (g_0, \dots, g_{n-1})$  – нулевая вершина куба следящей области. Пусть  $y = (y_0, \dots, y_{n-1})$  – нулевая вершина произвольной ячейки  $\alpha$ . Выразим координаты точки  $y$  через координаты точки  $g$ . Обозначим  $s = \frac{r}{K}$  – шаг сетки. Тогда

$$y_i = g_i + s\alpha_i \quad (i = 0, \dots, n-1). \quad (16)$$

Определим в качестве центральной ячейки куба ячейку  $\gamma$  с целочисленными координатами  $(\gamma_0, \dots, \gamma_{n-1})$ , где

<sup>2</sup> С помощью символа  $\div$  здесь обозначается целочисленное деление.

$$\gamma_0 = \dots = \gamma_{n-1} = \lfloor K/2 \rfloor. \quad (17)$$

Пусть  $q = (q_0, \dots, q_{n-1})$  – нулевая вершина центральной ячейки  $\gamma$ . Выразим координаты точки  $q$  через координаты точки  $g$ , используя формулу (16):

$$q_i = g_i + s\gamma_i \quad (i = 0, \dots, n-1). \quad (18)$$

### 3. Пересечение многогранника $M$ с ячейкой $\alpha$

Пусть  $y$  – нулевая вершина ячейки  $\alpha$ . Тогда область внутри ячейки  $\alpha$  (включая границы) задается системой из  $2n$  неравенств:

$$\left\{ \begin{array}{lll} -x_0 & \leq & -y_0 \\ & -x_1 & \leq -y_1 \\ & \dots & \dots \\ & & -x_{n-1} \leq -y_{n-1} \\ x_0 & \leq & y_0 + s \\ & x_1 & \leq y_1 + s \\ & \dots & \dots \\ & & x_{n-1} \leq y_{n-1} + s \end{array} \right. \quad (19)$$

Эта же система в матричной форме:

$$A_\alpha x \leq b_\alpha, \quad (20)$$

где (для  $n = 3$ )

$$A_\alpha = \begin{pmatrix} -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & -1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}, \quad b_\alpha = \begin{pmatrix} -y_0 \\ -y_1 \\ -y_2 \\ y_0 + s \\ y_1 + s \\ y_2 + s \end{pmatrix}. \quad (21)$$

Положим

$$A' = \begin{bmatrix} A \\ A_\alpha \end{bmatrix}, \quad b' = \begin{bmatrix} b \\ b_\alpha \end{bmatrix}. \quad (22)$$

Тогда пересечение многогранника  $M$  с ячейкой  $\alpha$  задается системой неравенств в матричной форме

$$A'x \leq b', \quad (23)$$

где  $A'$  – расширенная матрица размера  $(m + 2n) \times n$ ,  $b'$  – расширенный столбец свободных членов. Расширенный столбец  $b'$  в соответствии с формулой (22) имеет инвариантную часть  $b$ , не зависящую от координат нулевой вершины ячейки  $\alpha$ , и вариативную часть  $b_\alpha$ , зависящую от координат нулевой вершины ячейки  $\alpha$ . Элементы расширенной матрицы  $A'$  не зависят от координат нулевой вершины ячейки  $\alpha$ .

## 4. Реализация следящего алгоритма

В данном разделе описывается полная реализация следящего алгоритма в виде диаграмм деятельности UML.

### 4.1. Схема головной подпрограммы

Общая схема головной подпрограммы следящего алгоритма приведена на рис. 2. На шаге 1 выполняется подпрограмма *init* (см. раздел 4.2), выполняющая инициализацию

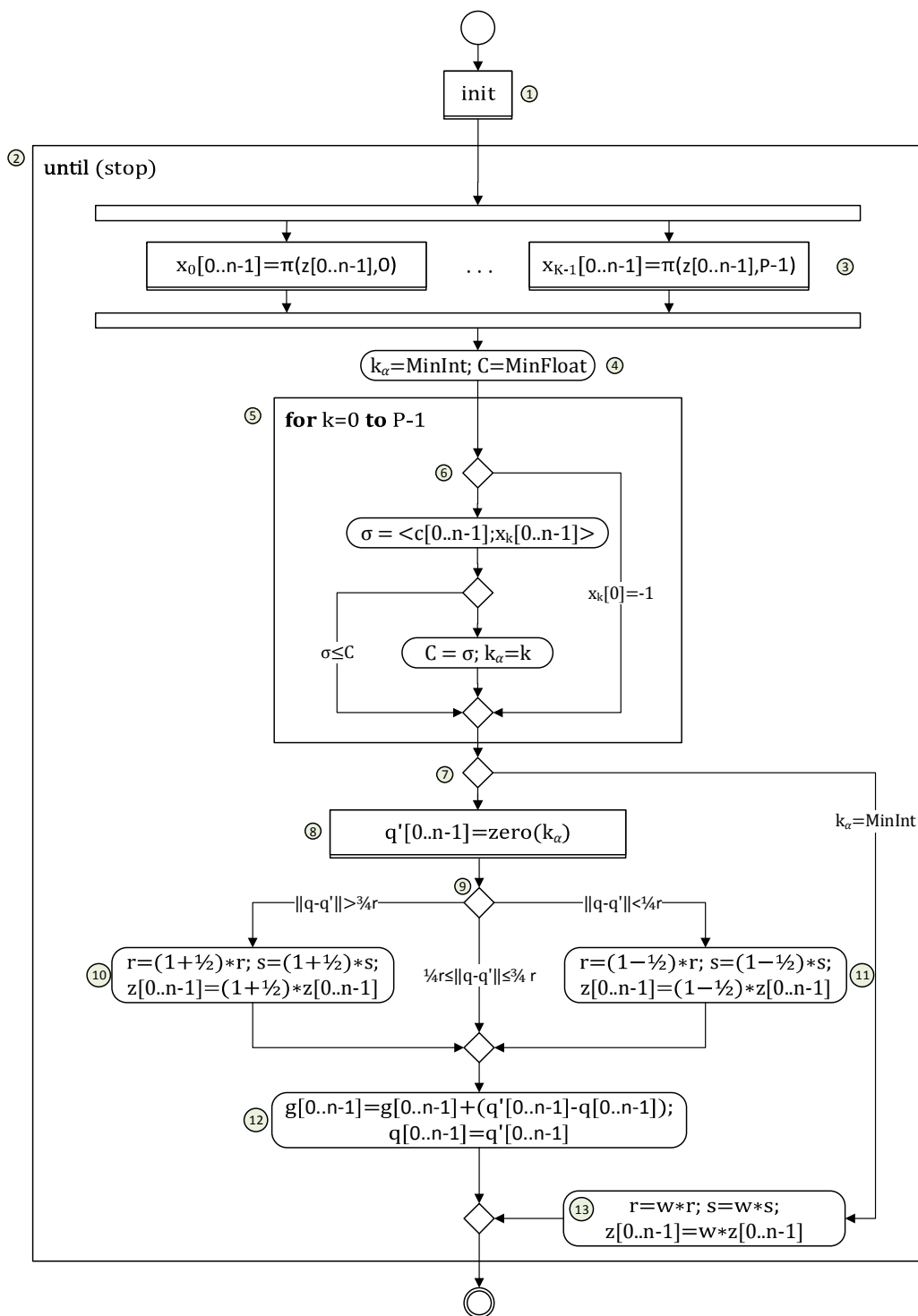


Рис. 2. Головная подпрограмма следящего алгоритма

переменных. Затем в цикле *until* с меткой 2 выполняется корректировка следящей области в соответствии с описанием идеи алгоритма в [11]. Одна итерация соответствует одной корректировке. Головная подпрограмма следящего алгоритма оформляется в виде независимого процесса, который выполняется до тех пор, пока переменная *stop* не примет значение *true* (истина). Начальную установку переменной *stop* в значение *false* (ложь) осуществляет головной процесс, соответствующий основной программе. Он же присваивает переменной *stop* значение *true*, когда вычислительный процесс нужно остановить. В качестве текущего приближения решения задачи (1) головная программа использует текущее значение нулевой вершины центральной ячейки  $q$ , координаты которой вычисляются по формуле (18).

В теле цикла *until* выполняются следующие действия. На шаге 3 организуется  $K$  параллельных потоков управления (нитей), которые независимо друг от друга вычисляют псевдопроекции из целевой точки  $z$  на пересечение  $i$ -той ячейки с многогранником  $M$  ( $i = 0, \dots, P - 1$ ). Напомним, что  $P$  равно количеству MPI-процессов, и в соответствии с формулой (5) равно количеству ячеек в кубической следящей области. Схема подпрограммы вычисления псевдопроекции детально описана в разделе 4.3.

В цикле *for* с меткой 5 для полученных на шаге 3 точек псевдопроекции  $x_0, \dots, x_{P-1}$  вычисляется номер  $k_\alpha$  ячейки, на которой достигается максимум  $C$  целевой функции. Для корректной работы цикла 5 переменным  $k_\alpha$  и  $C$  на шаге 4 присваиваются начальные значения *MinInt* и *MinFloat* соответственно. Значение *MinInt* соответствует минимальному машинному значению целого типа, *MinFloat* – минимальному машинному значению вещественного типа. Подпрограмма, вычисляющая точку  $x_k = (x_{k0}, \dots, x_{k,n-1})$  псевдопроекции точки  $z$  на пересечение многогранника  $M$  с ячейкой с номером  $k_\alpha$ , присваивает координате  $x_{k0}$  значение  $-1$  в том случае, когда точка  $x_k$  псевдопроекции не принадлежит многограннику  $M$ . Эта ситуация возникает в случае, когда пересечение многогранника  $M$  с ячейкой с номером  $k_\alpha$  является пустым. Если же  $x_k$  принадлежит многограннику, то в силу предположения о том, что все процессы находятся в положительной области координат (см. раздел 2), значение  $x_{k0}$  не может быть отрицательным. Указанное условие проверяется на шаге 6. Случаи  $x_{k0} = -1$  из рассмотрения исключаются. Если при выполнении цикла 5 оказывается, что ни одна из ячеек следящей области не имеет непустого пересечения с многогранником  $M$ , то в переменной  $k_\alpha$  сохраняется значение *MinInt*. Этот факт проверяется на шаге 7. В этом случае шаг сетки  $s$ , длина  $r$  ребра следящей области и координаты целевой точки  $z$  увеличиваются в  $w$  раз, где  $w$  – положительная константа, являющаяся параметром алгоритма (шаг 13 на рис. 2).

Если на шаге 7 выясняется, что  $k_\alpha \neq \text{MinInt}$ , значит найдена ячейка с номером  $k_\alpha$ , имеющая непустое пересечение с многогранником, на которой достигается максимум целевой функции. В этом случае на шаге 8 вычисляется вектор  $q'$ , представляющий нулевую вершину новой центральной ячейки следящей области. Схема подпрограммы вы-

числения нулевой вершины ячейки с порядковым номером  $k_\alpha$  приведена на рис. 3. Вычисления осуществляются с использованием формул (11), (15) и (16).

На шаге 9 (см. рис. 2) анализируется, насколько новая центральная ячейка далеко отстоит от предыдущей. Если расстояние между новой и старой центральными ячейками

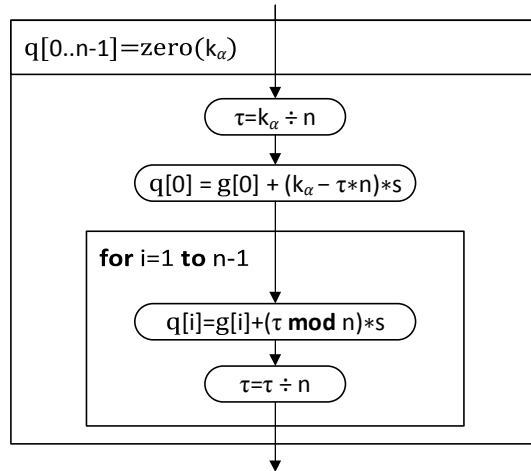


Рис. 3. Схема подпрограммы *zero* вычисления нулевой вершины ячейки с порядковым номером  $k_\alpha$

превышает  $\frac{3}{4}r$  (где  $r$  – длина ребра кубической следящей области), то длина ребра кубической следящей области  $r$ , шаг сетки  $s$  и координаты целевой точки  $z$  на шаге 10 увеличиваются в 1.5 раза. Если расстояние между новой и старой центральными ячейками меньше  $\frac{1}{4}r$ , то длина ребра кубической следящей области  $r$ , шаг сетки  $s$  и координаты целевой точки  $z$  на шаге 11 уменьшаются в 2 раза. Величина  $1/2$ , используемая в шагах 10 и 11, в общем случае является параметром алгоритма. Если же отклонение лежит в пределах от  $\frac{1}{4}r$  до  $\frac{3}{4}r$ , то корректировка значений  $r$ ,  $s$  и  $z$  не производится. Величины  $1/4$  и  $3/4$  также являются в общем случае параметрами алгоритма.

На шаге 12 следящая область сдвигается по вектору  $(q' - q)$ , и в качестве текущей нулевой вершины  $q$  центральной ячейки следящей области берется точка  $q'$ .

#### 4.2. Схема подпрограммы инициализации переменных *init*

Подпрограмма инициализации переменных *init* выполняет ввод исходных данных и инициализацию переменных. Схема подпрограммы *init* приведена на рис. 4. На шаге 1 вводятся значения переменных:  $n$  – размерность пространства решений;  $m$  – число неравенств в системе ограничений;  $R$  – начальное значение длины ребра следящей области, обеспечивающее покрытие многогранника  $M$ ;  $p$  – количество итераций при построении псевдопроекции, выполняемое между обновлениями входных данных (этот параметр используется в подпрограмме *dataChange* обновления исходных данных);  $K$  – количество ячеек в следящей области по одному измерению;  $u$  – размерность подвектора;  $L$  – число



независимых фейеровских итераций на подвекторах в подпрограмме вычисления псевдопроекции (см. рис. 5);  $T$  – масштабирующий коэффициент для вычисления координат целевой точки  $z$ . На шаге 2 проверяется условие (*assert*)  $n \bmod u = 0$ , означающее, что размерность пространства решений  $n$  кратно размерности подвектора  $u$ .

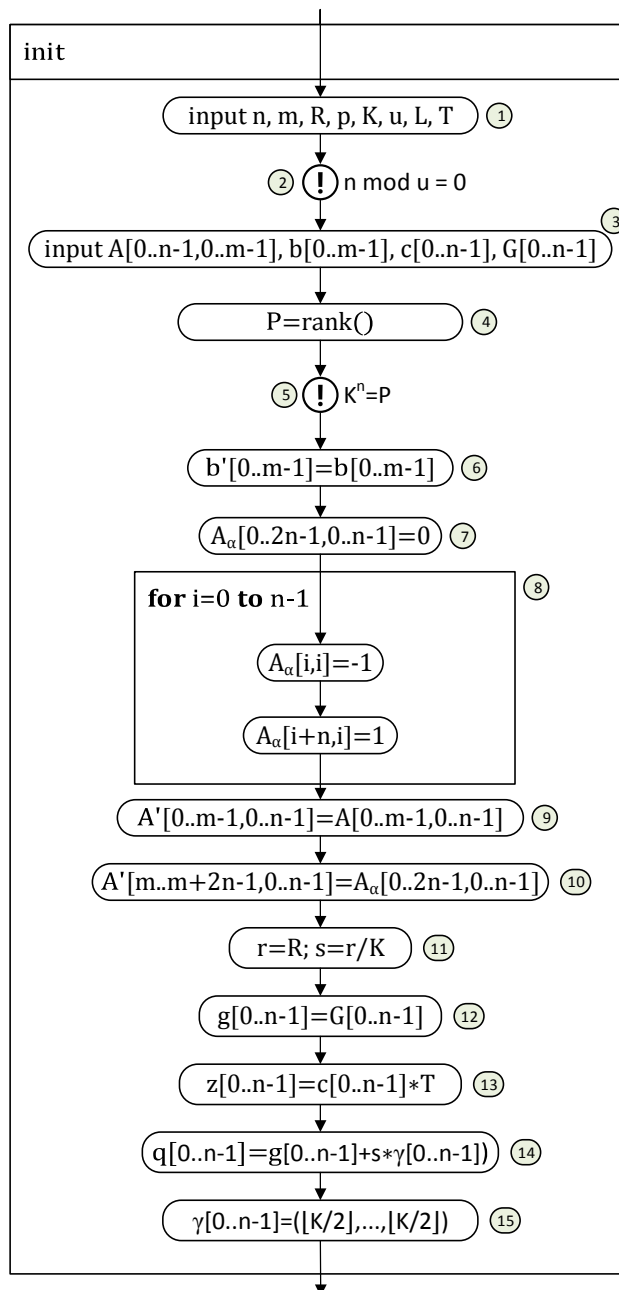


Рис. 4. Схема подпрограммы инициализации переменных *init*

На шаге 3 осуществляется ввод исходных данных задачи линейного программирования:  $A$  – матрица коэффициентов неравенств;  $b$  – столбец свободных членов;  $c$  – вектор коэффициентов целевой функции. Также здесь вводится вектор  $G$ , содержащий начальные координаты нулевой вершины кубической следящей области. Шаг 4 присваивает переменной  $P$  значение, равное количеству доступных MPI-процессов и вычисляемое с помощью системной функции *rank*. На шаге 5 проверяется условие (*assert*)  $K^n = P$ ,

означающее, что общее количество ячеек следящей области равно количеству MPI-процессов. На шаге 6 формируется инвариантная часть расширенного столбца  $b'$  свободных членов, определяемого по формуле (22). На шаге 7 происходит инициализация матрицы  $A_\alpha$  путем присваивания всем ее элементам нулевых значений. На шаге 8 определяются ненулевые элементы матрицы  $A_\alpha$  в соответствии с системой неравенств (19). На шагах 9 и 10 строится расширенная матрица  $A'$ , определяемая формулой (22). На шаге 11 в качестве начального значения длины  $r$  ребра следящей области определяется значение  $R$ , обеспечивающее покрытие многогранника  $M$ , и вычисляется значение  $s$  длины ребра ячейки. На шаге 12 в качестве начальной нулевой вершины  $g$  кубической следящей области берется точка  $G$ , определяющая такое положение следящей области, при котором она полностью покрывает многогранник  $M$ . На шаге 13 вычисляются координаты целевой точки  $z$  по формуле  $z = Tc$ . На шаге 14 вычисляется нулевая вершина  $q$  центральной ячейки следящей области по формуле (18). На последнем шаге вычисляется вектор  $\gamma$  начальных целочисленных координат центральной ячейки по формуле (17).

### 4.3. Схема подпрограммы вычисления псевдопроекции

На рис. 5 приведена схема подпрограммы вычисления псевдопроекции  $x = \pi(z, k_\alpha)$  из целевой точки  $z$  на пересечение многогранника  $M$  с ячейкой следящей области, имеющей порядковый номер  $k_\alpha$ , где  $k_\alpha$  вычисляется по формуле (6). Псевдопроекция вычисляется путем организации фейеровского процесса (4) (см. раздел 1). На шаге 1 выполняется инициализация переменных, необходимых для организации итерационного процесса. В качестве начального значения  $x_k$  берется точка  $z$ ; с помощью подпрограммы *zero* (см. рис. 3) вычисляется нулевая вершина  $y$  ячейки с номером  $k_\alpha$ ; по формуле (21) определяется вариативная часть  $b_\alpha$  расширенного столбца  $b'$  системы ограничений (23), получаемой при пересечении многогранника  $M$  с ячейкой  $\alpha$ . В цикле 2 вычисляется *normsq* – вектор квадратов норм строк расширенной матрицы  $A'$ :  $normsq_i = \|a'_i\|^2$ .

На шаге 3 организуется итерационный процесс вычисления псевдопроекции. Для обеспечения высокой масштабируемости процедуры вычисления псевдопроекции используется метод разбиения вектора  $x$  на  $h$  подвекторов размерности  $u$ , предложенный в работе [13]. Мы здесь предполагаем, что  $n = h \cdot u$ . На каждом  $v$ -том подвекторе делается  $L$  независимых итераций вида

$$(x_{vu}, \dots, x_{(v+1)u-1}) := (x_{vu}, \dots, x_{(v+1)u-1}) - \frac{\lambda}{m} \sum_{i=1}^m \frac{\max \left\{ \left\langle (a_{i,vu}, \dots, a_{i,(v+1)u-1}), (x_{vu}, \dots, x_{(v+1)u-1}) \right\rangle - b_i, 0 \right\}}{\|a_i\|^2} \cdot (a_{i,vu}, \dots, a_{i,(v+1)u-1}).$$

Указанная формула получается из формулы (3) путем ограничения действия фейеровского отображения  $\varphi$  на соответствующий подвектор. Подпрограмма *dataChange* вносит изменения в исходные данные с периодом в  $t$  секунд ( $t$  – положительное число, которое может принимать значения меньше единицы).

Итерационный процесс заканчивается, когда расстояние между двумя последними приближениями  $x$  и  $x'$  будет меньше  $\varepsilon$ . На четвертом шаге подпрограмма  $in$  (см. рис. 6) проверяет принадлежность найденной точки псевдопроекции  $x$  ячейке с номером  $k_\alpha$ . Если  $x$  не принадлежит ячейке с номером  $k_\alpha$ , то  $x[0]$  присваивается значение  $(-1)$ .

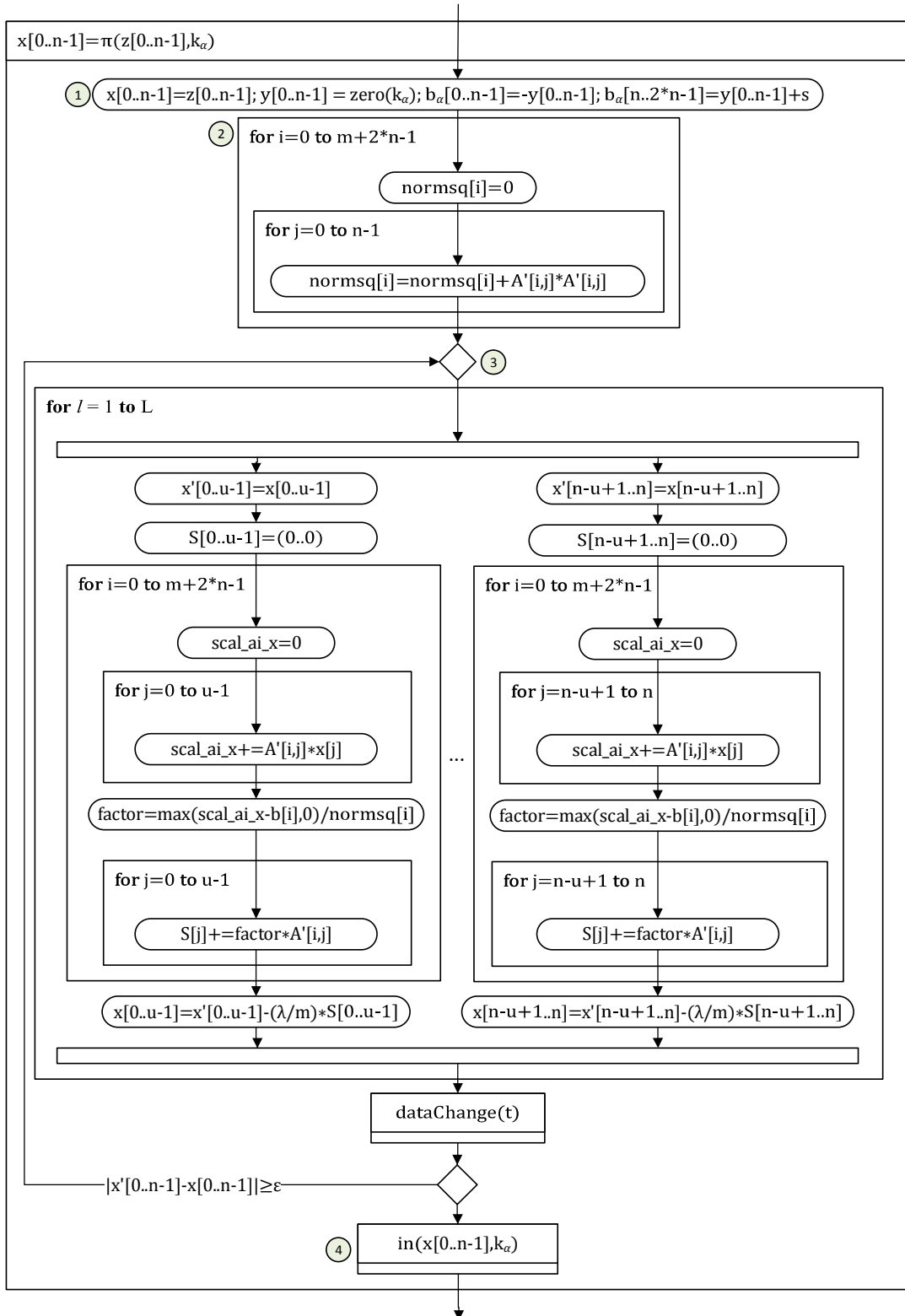
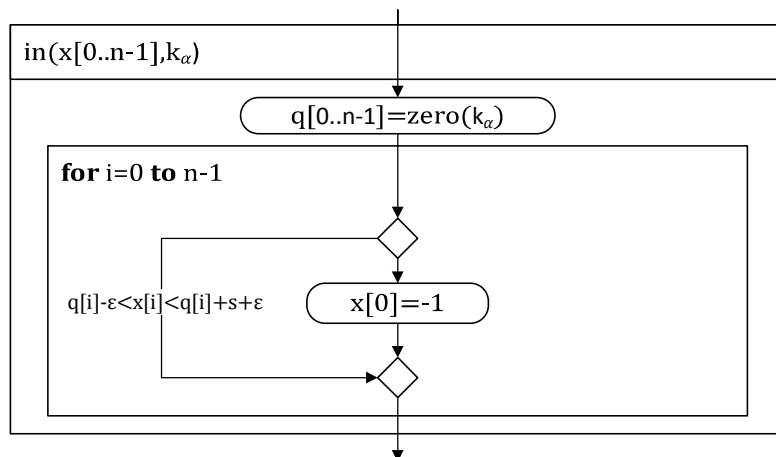


Рис. 5. Схема подпрограммы  $\pi$  вычисления псевдопроекции



**Рис. 6.** Схема подпрограммы *in*, проверяющей принадлежность точки  $x$  ячейке с номером  $k_\alpha$

Переменная  $s$  в подпрограмме *in* на рис. 6 задает длину ребра ячейки. Ее значение определяется на шаге 11 подпрограммы *init* (см. рис. 4). Константа  $\epsilon$  задает малое положительное число, позволяющее корректно обрабатывать приближенные значения.

## Заключение

В работе описана параллельная реализация следящего алгоритма для решения нестационарных задач большой размерности на кластерных вычислительных системах. Данный алгоритм использует подход, основанный на применении фейеровских отображений для построения псевдопроекции на многогранник. Приведена формальная постановка задачи, и дано описание фейеровского процесса. С помощью математических формул определены следящая область и пересечение многогранника, задаваемого системой ограничений, с произвольной ячейкой следящей области. В плане дальнейших исследований – реализация параллельного алгоритма на языке C++ с использованием технологий параллельного программирования MPI и OpenMP, и проведение вычислительных экспериментов на искусственных и реальных данных.

## Литература

1. Еремин И.И., Мазуров Вл.Д. Нестационарные процессы математического программирования. М.: Наука, 1979. 291 с.
2. Agmon S. The relaxation method for linear inequalities // Canad. J. Math. 1954. Vol. 6, No. 3. P. 382–392.
3. Motzkin T.S., Schoenberg J.J. The relaxation method for linear inequalities // Canad. J. Math. 1954. Vol. 6, No. 3. P. 393–404.
4. Еремин И.И., Попов Л.Д. Фейеровские процессы в теории и практике: обзор последних результатов // Известия высших учебных заведений. Математика. 2009. № 1. С. 44–65.
5. Дышаев М.М., Соколинская И.М. Представление торговых сигналов на основе адаптивной скользящей средней Кауфмана в виде системы линейных неравенств //

- Вестник Южно-Уральского государственного университета. Серия: Вычислительная математика и информатика. 2013. Т. 2, № 4. С. 103–108.
6. Ананченко И.В., Мусаев А.А. Торговые роботы и управление в хаотических средах: обзор и критический анализ // Труды СПИИРАН. 2014. № 3 (34). С. 178–203.
  7. Sokolinskaya I.M., Sokolinskii L.B. Parallel algorithm for solving linear programming problem under conditions of incomplete data // Automation and Remote Control. 2010. Vol. 71, No. 7. P. 1452–1460.
  8. Rechkalov T.V., Zymbler M.L. Accelerating Medoids-based Clustering with the Intel Many Integrated Core Architecture // Proceedings of the 9th International Conference on Application of Information and Communication Technologies (AICT'2015), October 14–16, 2015, Rostov-on-Don, Russia. IEEE, 2015. P. 413–417.
  9. Zymbler M.L. Best-match Time Series Subsequence Search on the Intel Many Integrated Core Architecture // Proceedings of the 19th East-European Conference on Advances in Databases and Information Systems, ADBIS 2015 (Poitiers, France, September 8–11, 2015). Lecture Notes in Computer Science. Springer, 2015. Vol. 9282. P. 275–286.
  10. Соколинская И.М., Соколинский Л.Б. Алгоритм решения нестационарных задач линейного программирования для кластерных вычислительных систем с многоядерными ускорителями // Параллельные вычислительные технологии (ПаВТ'2015): труды международной научной конференции. Челябинск: Издательский центр ЮУрГУ, 2015. С. 477–481.
  11. Sokolinskaya I., Sokolinsky L. Solving unstable linear programming problems of high dimension on cluster computing systems // 1st Russian Conference on Supercomputing Days 2015, RuSCDays 2015; Moscow; Russian Federation; 28 September 2015 through 29 September 2015. CEUR Workshop Proceedings. Vol. 1482, CEUR-WS.org 2015. P. 420–427.
  12. Еремин И.И. Фейеровские методы для задач выпуклой и линейной оптимизации. Челябинск: Изд-во ЮУрГУ, 2009. 200 с.
  13. Ершова А.В., Соколинская И.М. О сходимости масштабируемого алгоритма построения псевдопроекции на выпуклое замкнутое множество // Вестник Южно-Уральского государственного университета. Серия: Математическое моделирование и программирование. 2011. № 37(254), Вып. 10. С. 12–21.

Соколинская Ирина Михайловна, доцент, кафедра вычислительной математики, Южно-Уральский государственный университет (Челябинск, Российская Федерация), Irina.Sokolinskaya@susu.ru

Соколинский Леонид Борисович, д. ф.-м. н., профессор, проректор по информатизации, Южно-Уральский государственный университет (Челябинск, Российская Федерация), Leonid.Sokolinsky@susu.ru

*Поступила в редакцию 6 марта 2016 г.*

## IMPLEMENTATION OF PARALLEL PURSUIT ALGORITHM FOR SOLVING UNSTABLE LINEAR PROGRAMMING PROBLEMS

*I.M. Sokolinskaya*, South Ural State University, Chelyabinsk, Russian Federation

*L.B. Sokolinsky*, South Ural State University, Chelyabinsk, Russian Federation

The paper describes an implementation of the parallel pursuit algorithm for solving unstable linear programming problems of high dimension on cluster computing systems. This algorithm uses Fejer's mappings for building pseudo-projection on polyhedron. The algorithm tracks changes in input data and corrects the calculation process. This task is divided into set of independent subtasks, which can be processed in parallel. The UML activity diagrams describing the algorithm implementation are presented.

*Keywords:* unstable linear programming problem, Fejer's mappings, pursuit algorithm, UML activity diagrams, massive parallelism, cluster computing systems.

### FOR CITATION

Sokolinskaya I.M., Sokolinsky L.B. Implementation of Parallel Pursuit Algorithm for Solving Unstable Linear Programming Problems. *Bulletin of the South Ural State University. Series: Computational Mathematics and Software Engineering*. 2016. vol. 5, no. 2. pp. 15–29. (in Russian) DOI: 10.14529/cmse160202.

### References

1. Eremin I.I., Mazurov V.I.D. *Nestacionarnye protsessy matematicheskogo programmirovaniia* [Unstable Processes of Mathematical Programming]. Moscow, Nauka [Science], 1979. 291 p. (in Russian)
2. Agmon S. The Relaxation Method for Linear Inequalities. *Canad. J. Math.* 1954. vol. 6, no. 3. pp. 382–392.
3. Motzkin T.S., Schoenberg J.J. The Relaxation Method for Linear Inequalities. *Canad. J. Math.* 1954. vol. 6, no. 3. pp. 393–404.
4. Eremin I.I., Popov L.D. Feyerovskie protsessy v teorii i praktike: obzor poslednikh rezul'tatov [Fejer's Mappings in Theory and Practice: Review of the Latest Results]. *Izvestiya vysshikh uchebnykh zavedeniy. Matematika* [Proceedings of the Higher Educational Institutions. Mathematics]. 2009. no. 1. pp. 44–65. (in Russian)
5. Dyshaev M.M., Sokolinskaya I.M. Predstavlenie torgovykh signalov na osnove adaptivnoy skol'zyashchey sredney Kaufmana v vide sistemy lineynykh neravenstv [Representation of Trading Signals Based Kaufman Adaptive Moving Average as a System of Linear Inequalities]. *Vestnik Yuzhno-Ural'skogo gosudarstvennogo universiteta. Seriya: Vychislitel'naya matematika i informatika* [Bulletin of South Ural State University. Series: Computational Mathematics and Software Engineering]. 2013. vol. 2, no. 4. pp. 103–108. (in Russian)

6. Ananchenko I.V., Musaev A.A. Torgovye roboty i upravlenie v khaoticheskikh sredakh: obzor i kriticheskiy analiz [Trading Robots and Management in Chaotic Environments: an Overview and Critical Analysis]. *Trudy SPIIRAN* [SPIIRAS Proceedings]. 2014. no. 3 (34). pp. 178–203. (in Russian)
7. Sokolinskaya I.M., Sokolinskii L.B. Parallel Algorithm for Solving Linear Programming Problem Under Conditions of Incomplete Data. *Automation and Remote Control*. 2010. vol. 71, no. 7. pp. 1452–1460.
8. Rechkalov T.V., Zymbler M.L. Accelerating Medoids-based Clustering with the Intel Many Integrated Core Architecture. *Proceedings of the 9th International Conference on Application of Information and Communication Technologies (AICT'2015)*, October 14–16, 2015, Rostov-on-Don, Russia. IEEE, 2015. pp. 413–417.
9. Zymbler M.L. Best-match Time Series Subsequence Search on the Intel Many Integrated Core Architecture. *Proceedings of the 19th East-European Conference on Advances in Databases and Information Systems, ADBIS 2015 (Poitiers, France, September 8–11, 2015)*. *Lecture Notes in Computer Science*. vol. 9282. Springer, 2015. pp. 275–286.
10. Sokolinskaya I.M., Sokolinskiy L.B. Algoritm resheniya nestatsionarnykh zadach lineynogo programmirovaniya dlya klasternykh vychislitel'nykh sistem s mnogoyadernymi uskoritelyami [Algorithm for Solving Unstable Linear Programming Problems for Cluster Computing Systems with Manycore Accelerators]. *Parallel'nye vychislitel'nye tekhnologii (PaVT'2015): trudy mezhdunarodnoy nauchnoy konferentsii* [Parallel Computational Technologies (PCT'2015): Proceedings of the International Scientific Conference]. Chelyabinsk, Publishing of the South Ural State University, 2015. pp. 477–481. (in Russian)
11. Sokolinskaya I., Sokolinsky L. Solving Unstable Linear Programming Problems of High Dimension on Cluster Computing Systems. *1st Russian Conference on Supercomputing Days 2015, RuSCDays 2015; Moscow; Russian Federation; 28 September 2015 through 29 September 2015*. *CEUR Workshop Proceedings*. vol. 1482, CEUR-WS.org 2015. pp. 420–427. (in Russian)
12. Eremin I.I. *Fejerovskie metody dlya zadach lineynoy i vypukloj optimizatsii* [Fejer's Methods for Problems of Convex and Linear Optimization]. Chelyabinsk, Publishing of the South Ural State University, 2009. 200 p. (in Russian)
13. Ershova A.V., Sokolinskaya I.M. O skhodimosti masshtabiruemogo algoritma postroeniya psevdoproektsii na vypukloe zamknutoe mnozhestvo [About Convergence of Scalable Algorithm of Constructing Pseudo-Projection on Convex Closed Set]. *Vestnik Yuzhno-Ural'skogo gosudarstvennogo universiteta. Seriya: Matematicheskoe modelirovanie i programmirovaniye* [Bulletin of South Ural State University. Series: Mathematical simulation and programming]. 2011. no. 37(254), issue 10. pp. 12–21. (in Russian)

*Received March 6, 2016.*