УДК 681.3.06: 681.323(519.6)

ПАРАЛЛЕЛЬНЫЕ АЛГОРИТМЫ РЕШЕНИЯ БУЛЕВОЙ ЗАДАЧИ О РЮКЗАКЕ НА ОСНОВЕ СОРТИРОВКИ И ВИДОИЗМЕНЕНИЯ ФОРМУЛ ВИЕТА

Ромм Я.Е., Назарьянц Е.Г.

Таганрогский институт имени А.П. Чехова (филиал) ФГБОУ ВПО РГЭУ (РИНХ), Таганрог, e-mail: romm@list.ru

Предложены последовательный и два параллельных детерминированных алгоритма точного решения задачи об одномерном булевом рюкзаке. Первый параллельный алгоритм дает линейную оценку, при этом конструктивно задает каждое сочетание предметов с попутным вычислением веса и цены сочетания. Второй параллельный алгоритм выполняется за логарифмическое число шагов. Все варианты алгоритмов основаны на матричной модификации формул Виета для выражения коэффициентов многочлена по его корням, с помощью которой конструктивно задаются все сочетания общего вида из n предметов по l с попутным вычислением веса и цены каждого сочетания. В параллельном варианте используется алгоритм максимально параллельной сортировки подсчетом. Оценки временной сложности параллельных алгоритмов вида $T(2^{2n-3}) = O(n)$ и $T(2^{2n-1}) = O(\log_2 n)$, где в скобках левой части указано количество процессоров, представлены на модели неветвящихся параллельных программ без учета обмена.

Ключевые слова: задача об одномерном булевом рюкзаке, детерминированный алгоритм точного решения, временная сложность параллельного алгоритма, видоизменение формул Виета, параллельная модификация сортировки подсчетом

THE PARALLEL ALGORITHMS SOLUTION OF THE BOOLEAN PROBLEM ABOUT A KNAPSACK BASED ON THE SORTING AND THE MODIFICATIONS OF VIETA FORMULAS

Romm Y.E., Nazaryants E.G.

Anton Chekhov Taganrog State Institute (the branch) «Rostov State University of Economics (RINH)», Taganrog, e-mail: romm@list.ru

The consistent and two parallel deterministic algorithms of the exact solution of the problem of one-dimensional Boolean knapsack are offered. The first parallel algorithm provides a linear estimation, wherein structurally defines the each combination of items with a fair calculation of weight and price combination. The second parallel algorithm is performed by the logarithmic number of steps. All variants of algorithms are based on the matrix modification of Vieta formulas to express the coefficients of the polynomial by its roots, through which structurally are defined the all combinations of the general form of n objects on l with an obiter calculation of the weight and the price of each combination. The algorithm, that is maximally parallel to sorting by counting, is used in the parallel version. Assessments of the temporal complexity of parallel algorithms for the kinds $T(2^{2n-3}) = O(n)$ and $T(2^{2n-1}) = O(\log_2 n)$, where in the left part of the brackets the amount of processors is indicated, are shown on the models of non-branching parallel programs without taking into account the exchange.

Keywords: the task of one-dimensional Boolean knapsack, deterministic algorithm for the exact solution, the temporal complexity of the parallel algorithm, the modifications of Vieta formulas, the parallel modification of sorting by counting

Рассматривается традиционная постановка задачи о рюкзаке: имеется набор из n предметов, каждый предмет имеет вес w_i и цену p_i, $i = 1, 2, ..., n$, требуется собрать набор с максимальной ценой таким образом, чтобы он имел вес не больше w, где w – вместимость рюкзака [7]. Формальная запись постановки задачи:

$$\begin{cases} f(x) = \sum_{i=1}^{n} p_i x_i \to \max, \\ \sum_{i=1}^{n} w_i x_i \le w, \end{cases}$$

$$x_i \in \{0,1\}, \ 1 \le i \le n.$$

Требуется синтезировать детерминированные параллельные алгоритмы точного решения одномерной булевой задачи о рюкзаке с линейной и логарифмической оценками временной сложности. Временная сложность (кратко – время) $T(R)$, где R – количество процессоров, будет измеряться количеством последовательных шагов алгоритма на модели неветвящихся параллельных программ без учета обмена [8]. Искомые алгоритмы строятся с применением максимально параллельной сортировки на основе видоизменения формул Виета для восстановления коэффициентов многочлена через его корни.

Алгоритм выборки всех возможных сочетаний слагаемых для решения задачи о рюкзаке

Построение алгоритма опирается на структуру матричной формулы восстановления коэффициентов многочлена по его корням с точностью до выполнения опера-

ций, при этом непосредственно в исходном виде арифметические операции не выполняются. Коэффициенты многочлена

$$P_n(x) = d_n x^n + d_{n-1}x^{n-1} + d_{n-2}x^{n-2} + \ldots + d_1 x + d_0$$

выражаются через его корни в виде [3, 4]

$$\begin{pmatrix} d_n \\ d_{n-1} \\ \vdots \\ d_0 \end{pmatrix} = \underbrace{\begin{pmatrix} \left.\begin{matrix} 1 & 0 \\ x_{n-1} & 1 \\ 0 & x_{n-1} \\ 0 & 0 \\ \ldots & \ldots \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{matrix}\right\} n+1 & \begin{matrix} 0 & 0 \\ 0 & 0 \\ \vdots & \vdots \\ \vdots & \vdots \\ 0 & 0 \\ 1 & 0 \\ x_{n-1} & 1 \\ 0 & x_{n-1} \end{matrix} \\ \ldots & \end{pmatrix}}_{n} \times \underbrace{\begin{pmatrix} \left.\begin{matrix} 1 & 0 \\ x_{n-2} & 1 \\ 0 & x_{n-2} \\ 0 & 0 \\ \ldots & \ldots \\ 0 & 0 \\ 0 & 0 \end{matrix}\right\} n & \begin{matrix} 0 & 0 \\ 0 & 0 \\ \vdots & \vdots \\ 0 & 0 \\ 1 & 0 \\ x_{n-2} & 1 \\ 0 & x_{n-2} \end{matrix} \\ \ldots & \end{pmatrix}}_{n-1} \times \cdots \times \begin{pmatrix} 1 & 0 & 0 \\ x_2 & 1 & 0 \\ 0 & x_2 & 1 \\ 0 & 0 & x_2 \end{pmatrix} \times \begin{pmatrix} 1 & 0 \\ x_1 & 1 \\ 0 & x_1 \end{pmatrix} \times \begin{pmatrix} 1 \\ x_0 \end{pmatrix}, \quad (1)$$

где x_i – i-й корень многочлена, $i = 0, 1, ..., n - 1$, предполагается, что $d_n = 1$.

Формулы Виета для коэффициентов того же многочлена имеют вид

$$\left.\begin{aligned} & d_n = 1, \\ & d_{n-1} = -(x_0 + x_1 + x_2 + ... + x_{n-1}), \\ & d_{n-2} = (x_0 \cdot x_1) + (x_0 \cdot x_2) + ... + (x_0 \cdot x_{n-1}) + ... + (x_{n-2} \cdot x_{n-1}), \\ & d_{n-3} = -(x_0 \cdot x_1 \cdot x_2 + x_0 \cdot x_1 \cdot x_3 + ... + x_{n-3} \cdot x_{n-2} \cdot x_{n-1}), \\ & \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ & d_{n-l} = (-1)^{n-l} \cdot (x_0 \cdot x_1 \cdot x_2 \cdot x_3 ... \cdot x_{l-1} + ... + x_{n-l-1} \cdot ... \cdot x_{n-1}), \\ & \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ & d_0 = (-1)^n \cdot (x_0 \cdot x_1 \cdot x_2 \cdot ... \cdot x_{n-2} \cdot x_{n-1}). \end{aligned}\right\} \quad (2)$$

Левые части равенств (1) и (2) одинаковы, соответственно, равны правые части. В правых частях (2) – всевозможные сочетания корней, которые не повторяются, поэтому формула (1) порождает алгоритм генерации всех возможных сочетаний, если не принимать во внимание операции умножения и сложения. Если веса предметов в задаче о рюкзаке интерпретировать как корни многочлена, то из (1) следуют все возможные сочетания весов из n по m. В этой интерпретации произведение всех весов в сочетании заменяется на их сумму и не принимается во внимание знак слагаемых. Аналогично в (2) следует заменить знак умножения на знак сложения, знак суммы – на любой знак, обозначающий сочетание элементов (в таком качестве ниже выбрано логическое «ИЛИ»). С этими поправками по ходу умножения матриц (1) на момент окончания процесса получаются все сочетания из n по m предметов с заданными весами.

Общая схема решения задачи условно строится следующим образом [6].

Первый этап. Запись данных в виде произведения матриц (1).

Второй этап. Умножение матриц на текущем шаге записывается в виде [6]

$$\left.\begin{aligned} & d_{kk} = d_{(k-1)(k-1)}, \\ & d_{k(k-1)} = d_{(k-1)(k-2)} - d_{(k-1)(k-1)} x_{k-1}, \\ & d_{k(k-2)} = d_{(k-1)(k-3)} - d_{(k-1)(k-2)} x_{k-1}, \\ & \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ & d_{k(k-\ell)} = d_{(k-1)(k-\ell-1)} - d_{(k-1)(k-\ell)} x_{k-1}, \\ & \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ & d_{k0} = - d_{(k-1)0}\, x_{k-1}. \end{aligned}\right\},$$

$$k = 1, 2, ..., n. \quad (3)$$

Третий этап. Новое обозначение данных и операций. Последовательное вычисление сумм весов наборов предметов и цены наборов, отсеивание неперспективных наборов по весу с использованием сортировки. Для допустимых по весу наборов на текущем шаге определяется максимум стоимости.

В новом обозначении окончательный результат шагов преобразования примет вид

$$\left.\begin{array}{l} d_n \mapsto 1, \\ d_{n-1} \mapsto x_0 \vee x_1 \vee x_2 \vee \ldots \vee x_{n-1}, \\ d_{n-2} \mapsto (x_0 + x_1) \vee (x_0 + x_2) \vee (x_0 + x_3) \vee \ldots \vee (x_{n-2} + x_{n-1}), \\ d_{n-3} \mapsto (x_0 + x_1 + x_2) \vee (x_0 + x_1 + x_3) \vee \ldots \vee (x_{n-3} + x_{n-2} + x_{n-1}), \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ d_{n-l} \mapsto (x_0 + x_1 + x_2 + x_3 \ldots + x_{l-1}) \vee \ldots \vee (x_{n-l-1} + \ldots + x_{n-1}), \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ d_0 \mapsto (x_0 + x_1 + x_2 + \ldots + x_{n-2} + x_{n-1}). \end{array}\right\} \quad (4)$$

При этом в (4) условно не учитываются отсеянные на последовательных шагах сочетания предметов с недопустимым весом.

Четвертый этап. Для сохранившихся допустимых сочетаний на выходе схемы определяется сочетание с максимальной стоимостью.

Пример решения задачи по изложенной схеме приведен в [6].

Временная сложность последовательного алгоритма решения задачи о рюкзаке

Аналогично (4), текущий шаг (3) преобразуется к виду

$$\left.\begin{array}{l} d_{kk} \mapsto d_{(k-1)(k-1)}, \\ d_{k(k-1)} \mapsto d_{(k-1)(k-2)} \vee \left(d_{(k-1)(k-1)} + x_{k-1}\right), \\ d_{k(k-2)} \mapsto d_{(k-1)(k-3)} \vee \left(d_{(k-1)(k-2)} + x_{k-1}\right), \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ d_{k(k-\ell)} \mapsto d_{(k-1)(k-\ell-1)} \vee \left(d_{(k-1)(k-\ell)} + x_{k-1}\right), \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ d_{k0} \mapsto d_{(k-1)0} + x_{k-1}. \end{array}\right\}$$

$$k = 1, 2, \ldots, n. \quad (5)$$

Представление (5) выполняется путем записи и считывания элементов из памяти компьютера. В соответствии с принятой моделью в оценках временной сложности эти операции не учитываются.

Алгоритм (5) в рассматриваемом контексте выполняет генерацию сочетаний из k элементов по

$$d_{k(k-\ell)} \mapsto d_{(k-1)(k-\ell-1)} \vee \left(d_{(k-1)(k-\ell)} + x_{k-1}\right),$$

$$\ell = 1, 2, \ldots, k.$$

При этом коэффициент $d_{i(k-j)}$ является не числом, а обозначением набора всех сочетаний из i элементов по j.

Процесс подсчета веса каждого сочетания предметов на k-м шаге (5) заключается в однократном сложении x_{k-1} с весом сочетания из $k-1$ предметов по ℓ, подсчитанным на $(k-1)$-м шаге. Попутно подсчитывается суммарная цена предметов каждого сочетания и сравнивается с текущим максимумом по ℓ и по предшествующим значениям k. В результате получается текущее значение максимума цены на рассматриваемом шаге. Временная сложность последовательного выполнения k-го шага (5) составит

$$t_k = \left(1 + \sum_{\ell=1}^{k-1} C_{k-1}^{\ell}\right)\tau$$

или

$$t_k = \left(1 + 2^{k-1}\right)\tau, \quad (6)$$

где τ – время двух бинарных сравнений чисел, просуммированное с временем бинарного арифметического сложения по весу и по цене. Суммирование обеих частей равенства (6) по всем $k = 1, 2, \ldots, n$ даст оценку временной сложности последовательного детерминированного алгоритма подсчета веса и цены каждого допустимого сочетания с отсечением недопустимых весов, а также с последовательным формированием максимальной цены:

$$t \le \sum_{k=1}^{n} t_k = \sum_{k=1}^{n}\left(1 + 2^{k-1}\right)\tau = \left(2^n - 1 + n\right)\tau. \quad (7)$$

С учетом (7) на выходе алгоритма определяется искомый максимум цены среди допустимых сочетаний предметов по весам за время

$$T(1) \le \left(2^n + n\right)\tau. \qquad (8)$$

Предложение 1. Детерминированный алгоритм на основе (5) в последовательной форме дает точное решение задачи о рюкзаке с временной сложностью, оцениваемой из (8).

Алгоритм отличается от известных аналогов по построению линейной структурой, детерминированностью, инвариантностью, на каждом шаге отсекаются неперспективные наборы, что на практике может улучшать по времени поиск оптимального решения.

Параллельные алгоритмы решения задачи о рюкзаке

1. Шаг (5) выполняется параллельно по всем $\ell = 1, 2, ..., k$. Процесс подсчета веса каждого сочетания заключается в однократном сложении x_{k-1} с весом сочетания из $k - 1$ предметов по ℓ, подсчитанным на $(k - 1)$-м шаге. Попутно подсчитывается суммарная цена предметов каждого сочетания. Временная сложность параллельного выполнения k-го шага (5) оценивается временем однократного сложения в сочетании, включающем x_{k-1}, если количество процессоров совпадает с числом тех сочетаний из (5), в которые x_{k-1} входит, что составляет

$$1 + \sum_{\ell=1}^{k-1} C_{k-1}^{\ell} = 2^{k-1} + 1.$$

Для отсечения недопустимых весов применяется параллельная модификация сортировки подсчетом на основе матрицы сравнений [5] (для данного этапа рассматриваемого варианта алгоритма такая сортировка не является необходимой, но она окажется необходимой для параллельного нахождения максимальной цены допустимых сочетаний на k-м шаге; отсечение недопустимых весов можно было бы выполнять путем одновременного сравнения найденных весов сочетаний с заданной границей). Рассматриваемая сортировка для N элементов имеет временную сложность $T(R) = \tau$, где количество процессоров $R = \frac{1}{2}N\left(N+1\right)$. От всех отсортированных элементов параллельно вычитается граница допустимого веса, тогда смена неотрицательного элемента преобразованной последовательности на положительный отделяет всю совокупность допустимых весов от недопустимых. Аналогично сортируются цены всех допустимых на шаге весов. Максимальная цена располагается в конце отсортированного массива. Она сравнивается с текущим максимумом, определяемым по совокупности всех предшествующих $k - 1$ шагов, который аналогично формируется на каждом шаге данного вида. В результате сохраняется единичный порядок времени шага, при этом в оценке R следует положить $N = 2^{k-1} + 1$. Отсюда временная сложность k-го шага составит

$$T(R) = O(1),$$

где $R = \frac{1}{2}\left(2^{k-1}+1\right)\left(2^{k-1}+2\right) \sim 2^{2k-3}$. Число шагов меняется от 1 до n, поэтому суммарная оценка временной сложности данного параллельного алгоритма примет вид

$$T(R) = n\tau,$$

где τ из (6) увеличивается на время одного бинарного сравнения, а число процессоров берется максимальным по всем рассматриваемым шагам,

$$R \sim \max_{1 \le k \le n} 2^{2k-3} = 2^{2n-3}.$$

Таким образом, имеет место

Теорема 1. В рассматриваемой постановке задача о рюкзаке может быть точно решена с помощью детерминированного параллельного алгоритма на основе (5) и максимально параллельной сортировки с временной сложностью

$$T(2^{2n-3}) = n\tau, \qquad (9)$$

где τ включает время двух бинарных алгебраических сложений и трех бинарных сравнений.

Помимо отмеченных выше отличий алгоритм теоремы 1 включает свойство параллелизма. Как и последовательный алгоритм, он конструктивно выполняет генерацию всех сочетаний из k предметов по ℓ для всех $\ell = 1, 2, ..., k$ и всех $k = 1, 2, ..., n$.

2. Целесообразно рассмотреть параллельный алгоритм на основе непосредственно формул (4). Для ℓ-й строки (4), содержащей сочетания из n по ℓ в количестве C_n^{ℓ}, можно найти суммарный вес и суммарную цену каждого набора предметов параллельно по всем сочетаниям данной строки и одновременно по всем номерам строк $\ell = 1, 2, ..., n$. При этом вес (и цену) предметов в каждом сочетании ℓ-й строки можно найти по схеме сдваивания за время $(\log_2 \ell)$ τ на $\le \frac{1}{2}\ell$ процессорах. Всего в ℓ-й строке с учетом числа сочетаний потребуется

$\le \frac{1}{2}\ell C_n^\ell$ процессоров, вес и цену каждого набора в ℓ-й строке можно найти с временной сложностью

$$T_\ell\left(\frac{1}{2}\ell C_n^\ell\right)=\left(\log_2 \ell\right)\tau. \qquad (10)$$

Отбрасывание недопустимых сочетаний предметов выполняется путем одновременного сравнения найденных их весов с заданной границей, для чего заведомо достаточно число процессоров (10), при этом потребуется дополнительное время одного бинарного сравнения. Максимальное по всем ℓ время согласно (10) составит $T_n\left(\frac{1}{2}nC_n^n\right)=\left(\log_2 n\right)\tau$, а требуемое по всем параллельно обрабатываемым строкам число процессоров

$$R=\frac{1}{2}\sum_{\ell=1}^{n}\ell C_n^\ell \le \frac{1}{2}n\sum_{\ell=1}^{n}C_n^\ell = n2^{n-1}.$$

В итоге получится оценка:

$$T_1(n2^{n-1})=\left(\log_2 n\right)\tau. \qquad (11)$$

Максимальную цену допустимого набора можно получить параллельной сортировкой цен всех допустимых сочетаний, что, как и выше, возможно сделать с единичной оценкой времени на квадратичном от общего числа сочетаний количестве процессоров:

$$T_2(R) = \tau, \qquad (12)$$

где

$$R=\frac{2^{2n}-2^n}{2}\sim 2^{2n-1}. \qquad (13)$$

Поскольку $n\cdot 2^{n-1} \le \frac{2^{2n}-2^n}{2} \quad \forall n \ge 1$, то из (11)–(13) следует окончательная оценка временной сложности рассматриваемого параллельного алгоритма:

$$T\left(2^{2n-1}\right)=\left(\log_2 2n\right)\tau. \qquad (14)$$

Теорема 2. В данной постановке задача о рюкзаке может быть точно решена с помощью детерминированного параллельного алгоритма с временной сложностью (14), где τ – время двух бинарных сравнений и двух бинарных алгебраических сложений.

Согласно (14) алгоритм теоремы 2 улучшает оценку алгоритма теоремы 1 (9). Однако алгоритм теоремы 2, в отличие от алгоритма теоремы 1, не генерирует в конструктивной форме все сочетания. В этом смысле оценка (14) носит абстрактный характер. Вместе с тем все требуемые сочетания могут быть сгенерированы априори для комплекса возможных задач, и по индексам их элементов могут быть размещены предметы, относительно которых решается каждая конкретная задача о рюкзаке.

Заключение

В работе представлен последовательный и два параллельных детерминированных алгоритма точного решения задачи об одномерном булевом рюкзаке. Все разновидности алгоритмов основаны на модификации формул Виета для выражения коэффициентов многочлена по его корням, которая используется для параллельной генерации сочетаний. Кроме тог, использована максимально параллельная разновидность сортировки подсчетом по матрице сравнений. От известных работ [1, 2] по параллельным алгоритмам решения задачи о рюкзаке предложенный метод отличается по построению, а также детерминированным точным решением задачи с временной сложностью $O(n)$ либо $O(\log_2 n)$, в зависимости от варианта алгоритма.

Список литературы

1. Колпаков Р.М., Посыпкин М.А. Верхняя и нижняя оценки трудоемкости метода ветвей и границ для задачи о ранце // Дискретная математика. – 2010. – т. 22, Вып. 1. – С. 58–73.

2. Колпаков Р.М., Посыпкин М.А., Сигал И.Х. О нижней оценке вычислительной сложности одной параллельной реализации метода ветвей и границ // Автоматика и телемеханика. – 2010. – № 10. – С. 156–166.

3. Ромм Я.Е. Бесконфликтные и устойчивые методы детерминированной параллельной обработки: дис. ... д-ра техн. наук. – Таганрог: ТРТУ, 1999. – 546 с.

4. Ромм Я.Е. Локализация и устойчивое вычисление нулей многочлена на основе сортировки. II // Кибернетика и системный анализ. – 2007. – № 2. – С. 161–174.

5. Ромм Я.Е., Заика И.В. Численная оптимизация на основе алгоритмов сортировки с приложением к дифференциальным и нелинейным уравнениям общего вида // Кибернетика и системный анализ. – 2011. – № 2. – С. 165–180.

6. Ромм Я.Е., Назарьянц Е.Г. Детерминированный параллельный алгоритм решения задачи об одномерном булевом рюкзаке на основе сортировки и видоизменения формул Виета / ТИ имени А.П. Чехова (филиал) ФГБОУ ВПО РГЭУ (РИНХ). – Таганрог, 2015. – С. 45. Деп. в ВИНИТИ 18.02.2015, № 32-В2015.

7. Сигал И.Х, Иванова А.П. Введение в прикладное дискретное программирование. – М.: ФИЗМАТЛИТ, 2007. – 240 с.

8. Солодовников В.И. Верхние оценки сложности решения систем линейных уравнений // Теория сложности вычислений. I: Записки научных семинаров ЛОМИ АН СССР. – Л., 1982. – т. 118. – С. 159–187.

References

1. Kolpakov R.M., Posypkin M.A. Verhnjaja i nizhnjaja ocenki trudoemkosti metoda vetvej i granic dlja zadachi o rance // Diskretnaja matematika. 2010. t. 22, vypusk 1. pp. 58–73.

2. Kolpakov R.M., Posypkin M.A., Sigal I.H. O nizhnej ocenke vychislitelnoj slozhnosti odnoj parallelnoj realizacii met00oda vetvej i granic // Avtomatika i telemehanika. 2010. no. 10. pp. 156–166.

3. Romm Ja.E. Beskonfliktnye i ustojchivye metody determinirovannoj parallelnoj obrabotki // Dissertacija na soiskanie uchenoj stepeni doktora tehn. nauk. Taganrog. TRTU. 1999. 546 p.

4. Romm Ja.E. Lokalizacija i ustojchivoe vychislenie nulej mnogochlena na osnove sortirovki. II // Kibernetika i sistemnyj analiz. 2007. no. 2. pp. 161–174.

5. Romm Ja.E., Zaika I.V. Chislennaja optimizacija na osnove algoritmov sortirovki s prilozheniem k differencialnym i nelinejnym uravnenijam obshhego vida // Kibernetika i sistemnyj analiz. 2011. no. 2. pp. 165–180.

6. Romm Ja.E., Nazarjanc E.G. Determinirovannyj parallelnyj algoritm reshenija zadachi ob odnomernom bulevom rjukzake na osnove sortirovki i vidoizmenenija formul Vieta / TI imeni A.P. Chehova (filial) FGBOY VPO RGEY (RINX). Taganrog, 2015. pp. 45. Dep. v VINITI 18.02.2015, no. 32-V2015.

7. Sigal I.H, Ivanova A.P. Vvedenie v prikladnoe diskretnoe programmirovanie. M.: FIZMATLIT, 2007. pp. 240.

8. Solodovnikov V.I. Verhnie ocenki slozhnosti reshenija sistem linejnyh uravnenij // V kn.: Teorija slozhnosti vychislenij. I: Zapiski nauchnyh seminarov LOMI AN SSSR. L., 1982. t. 118. pp. 159–187.

Рецензенты:

Веселов Г.Е., д.т.н., директор Института компьютерных технологий и информационной безопасности, Инженерно-технологическая академия Южного федерального университета, г. Таганрог;

Карелин В.П., д.т.н., профессор, заведующий кафедрой прикладной математики и информационных технологий, Таганрогский институт управления и экономики, г. Таганрог.

Работа поступила в редакцию 01.04.2015.