ТЕХНИЧЕСКАЯ
КИБЕРНЕТИКА
№ 4 · 1981

УДК 62-50:519.283

ИСПОЛЬЗОВАНИЕ ДИНАМИЧЕСКОГО ПРОГРАММИРОВАНИЯ ПРИ ДВОЙСТВЕННОМ ПОДХОДЕ К РЕШЕНИЮ ЗАДАЧИ КАЛЕНДАРНОГО ПЛАНИРОВАНИЯ

КУКСА А. И., ЛАПТИН Ю. П.

Введение. Рассмотрим следующую задачу. Требуется выполнить совокупность из m работ. Работа i, $i=\overline{1, m}$, определена как последовательность из n_i операций, имеющих номера с 1 по n_i. Длительности операций являются целыми числами d_{ij}, где ij обозначает j-ю операцию работы i. Имеется l различных видов ресурсов, причем $r_{ij}^k \geqslant 0$ — количество k-го ресурса, используемого в процессе выполнения операции ij, R_{kt} — общее количество k-го ресурса в период времени t, время t измеряется целыми числами, $1 \leqslant t \leqslant T$. Начатые операции не разрешается прерывать. Выполнение операции состоит в предоставлении ей необходимого количества ресурсов на время d_{ij}.

Пусть t_{ij} обозначает время начала выполнения операции ij. Для каждой операции ij заданы ранние $\underline{t}_{ij}$ и поздние $\bar{t}_{ij}$ сроки начала выполнения, т. е. для t_{ij} должно выполняться условие

$$\underline{t}_{ij} \leqslant t_{ij} \leqslant \bar{t}_{ij}.$$

Положим $I_t = \{ij \mid t_{ij} \leqslant t \leqslant t_{ij}+d_{ij}\}$ — множество операций, выполняющихся в момент времени t; $f_{ij}=t_{ij}+d_{ij}$ — время завершения операции ij; $f_i = \max\limits_j f_{ij} = f_{in_i}$ — время завершения работы i. Технологические ограничения следования операций выражаются условиями вида

$$t_{ij} \geqslant t_{ik}+d_{ik},$$

если в последовательности операций работы i операция ij непосредственно следует за ik.

В каждый момент времени t должны выполняться ограничения по ресурсам:

$$\sum_{ij \in I_t} r_{ij}^k \leqslant R_{kt}, \quad k=\overline{1, l}, \quad t=\overline{1, T}.$$

Набор чисел $\{t_{ij}\}$, удовлетворяющий указанным условиям, определяет план выполнения работ — расписание. На множестве расписаний определена целевая функция

$$z = \sum_i g_i(f_i),$$

где $g_i(f_i)$ — неубывающие функции. Требуется найти расписание с минимальным значением z.

Описанную экстремальную задачу будем решать методом ветвей и границ, используя в качестве нижних границ решения двойственной задачи. Методы такого типа применительно к задачам календарного планирования

исследованы в [1–4]. Построим дальнейшее изложение по следующему плану: рассмотрим соответствующую задачу целочисленного линейного программирования, метод решения вспомогательной дискретной задачи, некоторые особенности организации для данной задачи алгоритма ветвей и границ и результаты численных экспериментов.

1. Двойственность и вычисление оценок. Выпишем линейную целочисленную формулировку задачи. Положим, что

$$x_{ij}{}^t = \begin{cases} 1, & \text{если операция } ij \text{ начинается в момент времени } t, \\ 0 & \text{в противном случае,} \end{cases}$$

$$c_i{}^t = g_i(t+d_{in_i}),$$

$T<\infty$ – общая длительность интервала планирования. Тогда экстремальная задача построения расписания состоит в том, чтобы найти

$$\min \sum_{i=1}^{m} \sum_{t=1}^{T} c_i{}^t x_{in_i}{}^t \tag{1.1}$$

при условиях

$$\sum_{t=\underline{t}_{ij}}^{\bar{t}_{ij}} x_{ij}{}^t = 1, \quad i=\overline{1,m}, j=\overline{1,n_i}, \tag{1.2}$$

$$\sum_{ij} \sum_{\tau=\max(0,t-d_{ij}+1)}^{t} r_{ij}{}^k x_{ij}{}^\tau \leqslant R_{kt}, \quad k=\overline{1,l}, \quad t=\overline{1,T}, \tag{1.3}$$

$$\sum_t t x_{ij}{}^t + d_{ij} \leqslant \sum_t t x_{i(j+1)}{}^t, \quad i=\overline{1,m}, \quad j=\overline{1,n_i-1}, \tag{1.4}$$

$$x_{ij}{}^t = 0 \vee 1 \quad \forall ijt. \tag{1.5}$$

Набор $\{x_{ij}{}^t\}$ назовем псевдорешением задачи (1.1)–(1.5), если он удовлетворяет ограничениям (1.2), (1.4), (1.5). Множество всех псевдорешений обозначим X.

Рассмотрим функцию Лагранжа

$$L(x,u) = \sum_i \sum_t c_i{}^t x_{in_i}{}^t + \sum_{kt} u_{kt} \left(\sum_{ij} \sum_{\tau=\max(0,t-d_{ij}+1)} r_{ij}{}^k x_{ij}{}^\tau - R_{kt} \right). \tag{1.6}$$

Пусть

$$w(u) = \min_{x \in X} L(x,u). \tag{1.7}$$

Так как X – конечное множество, то $w(u)$ – кусочно-линейная вогнутая функция.

Положим

$$w^* = \max_{u \geqslant 0} w(u). \tag{1.8}$$

Известно [5], что если v^* – оптимальное значение задачи (1.1)–(1.5), то $w^* \leqslant v^*$.

Рассмотрим задачу (1.7). Набор $x_{(i)} = \{x_{ij}{}^t\}$, $j=\overline{1,n_i}$, $t=\overline{1,T}$, назовем расписанием i-й последовательности операций, если он удовлетворяет ограничениям (1.2), (1.4), (1.5). Множество всех таких наборов для i-й последовательности обозначим $\overline{X}_i$. Нетрудно видеть, что $X = \overline{X}_1 \times \ldots \times \overline{X}_m$.

Определим

$$\pi_i(x_{(i)}, u) = \sum_t c_i^t x_{in_i}^t + \sum_{kt} u_{kt} \sum_{j=1}^{n_i} \sum_{\tau=\max(0, t-d_{ij}+1)}^{t} r_{ij}^k x_{ij}^\tau, \qquad (1.9)$$

$$x_{(i)} \in \overline{X}_i, \quad i=\overline{1, m}.$$

С учетом (1.9) задачу (1.7) можно представить в следующем виде:

$$w(u) = \min_{x_{(1)}, \ldots, x_{(m)}} \left\{ \sum_{i=1}^{m} \pi_i(x_{(i)}, u) - \sum_{kt} u_{kt} R_{kt} \right\} =$$

$$= \sum_{i=1}^{m} \min_{x_{(i)} \in \overline{X}_i} \pi_i(x_{(i)}, u) - \sum_{kt} u_{kt} R_{kt}.$$

Таким образом, для решения задачи (1.7) достаточно для каждой последовательности операций найти $\pi_i(u) = \min\limits_{x_{(i)} \in \overline{X}_i} \pi_i(x_{(i)}, u)$ или, более подробно, решить задачу (индекс i опущен):

$$\pi(u) = \min \left\{ \sum_t c^t x_n^t + \sum_{kt} u_{kt} \sum_j \sum_{\tau=\max(0, t-d_j+1)}^{t} r_j^k x_j^\tau \right\}, \qquad (1.10)$$

$$\sum_{t=\underline{t}_j}^{\overline{t}_j} x_j^t = 1, \quad j=\overline{1, n}, \qquad (1.11)$$

$$\sum_t t x_j^t + d_j \leqslant \sum_t t x_{j+1}^t, \quad j=\overline{1, n-1}, \qquad (1.12)$$

$$x_j^t = 0 \vee 1, \quad j=\overline{1, n}, \quad t=\overline{1, T}. \qquad (1.13)$$

Опишем метод динамического программирования для решения задачи (1.10)–(1.13).

Множеству решений задачи поставим в соответствие сеть (V, E, H), вершины которой соответствуют различным моментам начала выполнения операций. Сеть (V, E, H) состоит из $n+2$ слоев и удовлетворяет следующим условиям.

1. Множество вершин $V = \{0, \bigcup\limits_{j=1}^{n} V_j, 1\}$. Множество V_j, $j=\overline{1, n}$, состоит из всех вершин v_j^t таких, что $\underline{t}_j \leqslant t \leqslant \overline{t}_j$, t – целое.

2. Дугами связаны только те вершины, которые принадлежат соседним слоям V_j и V_{j+1}, $j=\overline{0, n}$:

а) вершина 0 (слой 0) связана дугами $(0, v_1^t)$ со всеми вершинами первого слоя, $v_1^t \in V_1$, $t=\underline{t}_1, \ldots, \overline{t}_1$;

б) пара вершин $v_j^{t_1} \in V_j$ и $v_{j+1}^{t_2} \in V_{j+1}$ из соседних слоев связана дугой $(v_j^{t_1}, v_{j+1}^{t_2})$, если $t_1 + d_j \leqslant t_2$;

в) вершина 1 (слой $(n+1)$) связана дугами $(v_n^t, 1)$ со всеми вершинами n-го слоя, $v_n^t \in V_n$, $t=\underline{t}_n, \ldots, \overline{t}_n$.

3. Веса, сопоставленные вершинам (множество H), подсчитываются следующим образом:

$$h(0) = h(1) = 0,$$

$$h(v_j^t) = \sum_{\tau=t}^{t+d_j} \sum_{k=1}^{l} r_j^k u_{k\tau}, \quad j=\overline{1, n-1},$$

$$h(v_n^t)=\sum_{\tau=t}^{t+d_n}\sum_{k=1}^{l} r_n^k u_{k\tau}+c^t.$$

Нетрудно видеть, что каждому пути $(0, v_1^{\tilde{t}_1}, \ldots, v_n^{\tilde{t}_n}, 1)$ из вершины 0 в вершину 1 в сети (V, E, H), если он существует, соответствует допустимое решение $\tilde{x}=\{\tilde{x}_j^t\}$ задачи (1.10)–(1.13)

$$\tilde{x}_j^t=\begin{cases}1, & \text{если } t=\tilde{t}_j,\\ 0 & \text{в противном случае.}\end{cases}$$

При этом $\pi(\tilde{x}, u)=\sum_{j=1}^{n} h(v_j^{\tilde{t}_j})$. Задача $\pi(u)=\min_{x\in X}\pi(x, u)$ сводится, таким образом, к нахождению в сети (V, E, H) кратчайшего пути, соединяющего вершины 0 и 1. Учитывая простую структуру сети, алгоритм динамического программирования для этой задачи можно организовать таким образом, что его трудоемкость будет $\sim nT$. Поскольку эта задача хорошо изучена, мы не будем останавливаться на особенностях соответствующей программы.

Для решения задачи (1.8) можно использовать методы обобщенного градиентного спуска [5, 6], на которых мы также не будем останавливаться.

2. Особенности организации алгоритма ветвей и границ 2.1. Генерирование новых подзадач. Как и ранее, $\underline{t}_{ij}$, $\bar{t}_{ij}$ обозначают соответственно ранние и поздние сроки начала выполнения операции ij (для исходной задачи $\underline{t}_{ij}^0$, $\bar{t}_{ij}^0$). Будем говорить, что сроки $\underline{t}_{ij}$, $\bar{t}_{ij}$ скорректированы, если

$$\underline{t}_{i1}\geqslant 1, \quad t_{in_i}+d_{in_i}\leqslant T, \quad i=\overline{1, m},$$

$$\underline{t}_{i(j-1)}+d_{i(j-1)}\leqslant \underline{t}_{ij}\leqslant \bar{t}_{i(j+1)}-d_{ij}, \quad i=\overline{1, m}, \quad j=\overline{2, n_i-1}.$$

Каждая вершина дерева ветвления (подзадача) идентифицируется набором $\{\underline{t}_{ij}, \bar{t}_{ij}\}$, $i=\overline{1, m}$, $j=\overline{1, n_i}$. Резервом времени операции ij назовем разность $W_{ij}=\bar{t}_{ij}-\underline{t}_{ij}$.

Пусть для ветвления выбрана подзадача $\{\underline{t}_{ij}, \bar{t}_{ij}\}$ и операция $i'j'$ имеет максимальный резерв времени $W_{i'j'}$.

Полагая

$$\underline{t}_{i'j'}^1=\underline{t}_{i'j'}, \ \bar{t}_{i'j'}^1=\left[\frac{\underline{t}_{i'j'}+\bar{t}_{i'j'}}{2}\right],$$

$$\underline{t}_{i'j'}^2=\left]\frac{\underline{t}_{i'j'}+\bar{t}_{i'j'}}{2}\right[, \quad \bar{t}_{i'j'}^2=\bar{t}_{i'j'}$$

и корректируя ранние и поздние сроки остальных операций, получаем две новые подзадачи $\{\underline{t}_{ij}^1, \bar{t}_{ij}^1\}$ и $\{\underline{t}_{ij}^2, \bar{t}_{ij}^2\}$. Здесь $[x]$ – наибольшее целое, не превышающее x, $]x[$ – наименьшее целое, большее x.

2.2. Пересчет оценок. Для решения задачи (1.8) используется метод обобщенного градиентного спуска с растяжением пространства [6]. Обобщенный градиент, как легко видеть, равен

$$G_{kl}(u)=\sum_{ij}\sum_{\tau=\max(0, t-d_{ij}+1)}^{t} r_{ij}^k x_{ij}^{\tau}(u)-R_{kl},$$

где $\{x_{ij}^t(u)\}$ – решение задачи (1.7) при заданном u.

Обозначим u_0 начальные значения множителей Лагранжа. Тогда для исходной задачи $u_0=0$; в дальнейшем при вычислении оценки порожденной подзадачи полагаем $u_0=\bar{u}$, где $\bar{u}$ – наилучшее значение множителей для порождающей подзадачи.

Число итераций градиентного спуска выбирается сравнительно неболь-

шим (~50) ввиду большой трудоемкости вычислений значений функции $w(u)$ и градиента $G_{kt}(u)$.

Пусть вершина $\{\underline{t}_{ij}', \bar{t}_{ij}'\}$ – непосредственный потомок вершины $\{\underline{t}_{ij}, \bar{t}_{ij}\}$, полученный при уменьшении резерва времени операции $i'j'$. Рассмотрим решение $\{x_{ij}^t(\bar{u})\}$ ($\{\tilde{x}_{ij}^t(\bar{u})\}$) задачи (1.6), (1.7), соответствующей вершине $\{\underline{t}_{ij}, \bar{t}_{ij}\}$ (соответственно $\{\underline{t}_{ij}', \bar{t}_{ij}'\}$) при $u=\bar{u}$. Очевидно, $\underline{t}_{ij}'=\underline{t}_{ij}$, $\bar{t}_{ij}'=\bar{t}_{ij}$, если $i\neq i'$, откуда следует $\tilde{x}_{ij}^t(\bar{u})=x_{ij}^t(\bar{u})$ при $i\neq i'$, т. е. при вычислении оценки $w(\bar{u})$ для новой вершины $\{\underline{t}_{ij}', \bar{t}_{ij}'\}$ достаточно найти решение задачи (1.10)–(1.13) только для той последовательности операций, для которой изменились ранние и поздние сроки.

Таким образом, для уменьшения трудоемкости алгоритма в целом представляется целесообразным вычислять оценку $w(u)$ для некоторой части вершин при фиксированных значениях множителей Лагранжа.

Пересчет множителей (решение задачи (1.8)) делается для всех вершин дерева ветвления, ранг которых (расстояние от корня дерева) равен pn, $n=0, 1, \ldots$, где p – параметр алгоритма (p – целое число).

2.3. Построение допустимого расписания. Исследуемая нами задача является полиномиально полной в смысле Кука – Карпа, поэтому следует ожидать, что поиск точного решения задачи потребует экспоненциального объема вычислений. Синтезируя алгоритм ветвей и границ, будем стремиться поэтому к получению наилучшего по функционалу решения в заданное время счета. Технически эта идея может быть реализована посредством организации внутри общей схемы алгоритма ветвей и границ специальной процедуры пересчета верхних границ. Верхнюю границу в исследуемой нами задаче дает любой эвристический алгоритм ее решения. Описываемый ниже алгоритм оперирует информацией, содержащейся в псевдорешениях подзадач, т. е. в результатах вычисления нижних границ. Опишем соответствующий алгоритм построения допустимых расписаний.

Расписание $\{t_{ij}\}$ будем называть активным, если не существует другого расписания $\{t_{ij}'\}$ такого, что $t_{ij}'\neq t_{ij}$ по крайней мере для одной операции и $t_{ij}'\leqslant t_{ij}$ для всех ij. Очевидно, что в множестве активных расписаний содержится хотя бы одно оптимальное.

Пусть задан набор приоритетов операций $\{\sigma_{ij}\}$ такой, что $\sigma_{ij}\neq\sigma_{i'j'}$ при $i'j'\neq ij$, $\sigma_{ij}<\sigma_{i'j'}$, если операция ij предшествует операции $i'j'$. По набору $\{\sigma_{ij}\}$ нетрудно построить некоторое активное расписание. Для этого достаточно последовательно «наращивать» начальный отрезок расписания, выбирая операции в порядке возрастания значений σ_{ij} и «сдвигая» их максимально влево с учетом ранних сроков исходной задачи и ограничений (1.3), (1.4).

Такая процедура имеет трудоемкость $\sim T\sum_{i=1}^{m} n_i$.

Допустимое решение строится всякий раз, когда пересчитываются множители Лагранжа (после приближенного решения задачи (1.8)). Пусть наилучшая оценка $w(u)$ вершины $\{\underline{t}_{ij}, \bar{t}_{ij}\}$ получена при $u=\bar{u}$, $\{x_{ij}^t(\bar{u})\}$ – соответствующее псевдорешение. Тогда в качестве набора приоритетов операций выбирается набор $\{\sigma_{ij}\}$, удовлетворяющий следующим условиям:

1) $\sigma_{ij}\neq\sigma_{i'j'}$, если $ij\neq i'j'$;

2) $\sigma_{ij}>\sigma_{i'j'}$, если $\sum_t t x_{ij}^t(\bar{u}) > \sum_t t x_{i'j'}^t(u)$;

3) в случае $\sum_t t x_{ij}^t(\bar{u}) = \sum_t t x_{i'j'}^t(u)$ полагаем $\sigma_{ij}>\sigma_{i'j'}$, если

$$\sum_t c_i^t x_{in_i}^t(\bar{u}) < \sum_t c_{i'}^t x_{i'n_{i'}}^t.$$

2.4. Для описания стратегии выбора вершины для ветвления определим приоритет W вершины $\{\underline{t}_{ij}, \bar{t}_{ij}\}$

$$W=w+\alpha\sum_{ij}(\bar{t}_{ij}-\underline{t}_{ij}),$$

где w — оценка вершины $\{\underline{t}_{ij}, \bar{t}_{ij}\}$, α — параметр стратегии.

Каждый раз для ветвления выбирается вершина, имеющая наименьший приоритет. Если в некоторый момент окажется, что $\widetilde{W}>\tilde{v}^0$, где $\widetilde{W}$ — приоритет наилучшей вершины, $\tilde{v}^0$ — текущий рекорд, полагаем $\alpha:=\alpha q$, $0\leqslant q\leqslant 1$.

Вычислительный процесс оканчивается, если истекло время, выделенное для решения задачи или получено оптимальное решение.

Нетрудно видеть, что если $q=1$, α — достаточно большое число, то такая стратегия ветвления эквивалентна одностороннему обходу дерева ветвления, при $\alpha=0$ получаем поиск по наилучшей оценке.

2.5. Описанный алгоритм реализован на языке ПЛ-1 для операционной системы ОС ЕС ЭВМ. Для запоминания дерева ветвления (информации о подзадачах) используется внешняя память (диски).

Вычислительные эксперименты проводились на ЭВМ ЕС-1040 для серии из шести тестовых задач. Каждая задача просчитывалась при различных значениях параметров алгоритма. Время счета одного варианта — 10 *мин*.

Тестовые задачи составлялись из последовательностей операций двух типов (в каждой последовательности четыре операции, все операции потребляют один вид ресурса):

1) $d_1=3$, $d_2=1$, $d_3=1$, $d_4=2$; $r_1=3$, $r_2=4$, $r_3=6$, $r_4=3$;

2) $d_1=3$, $d_2=1$, $d_3=3$, $d_4=2$; $r_1=4$, $r_2=6$, $r_3=2$, $r_4=3$.

Каждая задача состояла из пяти последовательностей, функции $g_i(f_i)$, $i=\overline{1,5}$, и интервал планирования T для всех задач одинаковы: $T=30$; $g_i(f_i)=\max(0, f_i-s_i)$, где $s_i=1+2\cdot i$, $i=\overline{1,5}$. Таким образом, каждая задача определяется набором последовательностей и количеством наличного ресурса ($R_t=R$ — постоянная). Характеристики каждой задачи приведены ниже:

Номер задачи	1	2	3	4	5	6
Типы последовательностей	1	1	2	2	1 — 3-я — тип 1 4-, 5-я — тип 2	
Количество наличного ресурса	9	8	9	8	9	8

Результаты счета приведены в таблице. Обозначения: f — наилучший рекорд, N — число построенных подзадач, N^* — число построенных подзадач к моменту последнего уточнения рекорда, α, q, p — параметры алгоритма.

Из результатов эксперимента следует, что для решения рассмотренных задач наиболее целесообразно применять смешанную стратегию ($\alpha=1$, $q=0.8$) с параметром $p>1$. Необходимо отметить также, что при $p>2$ все стратегии дают приблизительно одинаковые решения.

Анализ работы алгоритма показал, что в ходе вычислений оценка снизу оптимального решения растет сравнительно медленно и по окончании работы алгоритма имеет близкие значения для всех вариантов счета одной задачи. Это можно объяснить относительно малым числом построенных подзадач. Приведем оценки оптимальных решений (w_i — наилучшая оценка i-й задачи): $w_1\approx 32.3$, $w_2\approx 36.4$, $w_3\approx 46.5$, $w_4\approx 51.1$, $w_5\approx 40.3$, $w_6\approx 44.3$.

Заключение. Как отмечено выше, использованию двойственности в сетевых задачах теории расписаний посвящены работы [1—4]. Оставаясь в пределах тех же принципов, в настоящей работе мы усовершенствовали

Таблица

		Задача 1				Задача 2				Задача 3			
p		1	2	3	4	1	2	3	4	1	2	3	4
N		54	110	160	200	54	110	160	200	57	105	140	200
Односторонний обход дерева ветвления	f	39	39	38	39	53	57	57	51	60	58	60	54
	N^*	39	39	60	70	44	41	44	60	6	71	4	116
Поиск по наилучшей оценке	f	39	39	39	39	62	58	51	51	66	61	59	59
	N^*	42	42	46	46	44	47	121	110	2	102	95	112
$\alpha = 1$	f	39	39	38	38	58	51	51	51	57	57	57	57
$q = 0.8$	N^*	42	54	109	108	6	68	72	94	43	95	113	188

		Задача 4				Задача 5				Задача 6			
p		1	2	3	4	1	2	3	4	1	2	3	4
N		57	105	140	200	54	105	180	240	54	105	180	240
Односторонний обход дерева ветвления	f	65	61	61	61	51	51	50	51	58	56	56	56
	N^*	34	78	98	96	6	10	65	12	6	54	39	63
Поиск по наилучшей оценке	f	61	61	61	61	56	52	51	50	56	56	56	56
	N^*	50	50	63	96	1	77	176	234	34	34	26	28
$\alpha = 1$	f	61	61	61	61	51	51	50	50	56	56	55	56
$q = 0.8$	N^*	38	38	41	38	6	10	135	73	42	51	18	36

и сделали более эффективными наиболее часто повторяющиеся внутренние процедуры алгоритма, а именно решение задачи минимизации по x (задача (1.7)) и задачи максимизации по u (задача (1.8)). Задача дискретного программирования (1.7) сведена к простейшей задаче о кратчайшем пути, в задаче (1.8) предлагается использовать мощные и хорошо зарекомендовавшие себя современные методы негладкой оптимизации. Следует ожидать, что указанные нововведения приводят к сокращению времени счета. Более точных сравнений на этот счет мы не в состоянии сделать ввиду недоступности для нас тестовых задач или же невоспроизводимости численных результатов из публикаций [1—4]. Придавая большое значение указанным вопросам, мы полностью приводим как использованные нами задачи, так и результаты экспериментов.

ЛИТЕРАТУРА

1. *Fisher M. L.* Optimal Solution of Scheduling Problems Using Lagrange Multipliers. Part I.— Operat. Res., 1973; v. 21, № 5.
2. *Fisher M. L.* Optimal Solution of Scheduling Problems Using Lagrange Multipliers. Part II.— Lect. Notes Econom. and Math. Syst., 1973, v. 86.
3. *Демин В. К., Чеботарев А. С.* Оптимизация обслуживания детерминированного потока требований. I.– Изв. АН СССР. Техн. кибернет., 1976, № 4.
4. *Демин В. К., Чеботарев А. С.* Оптимизация обслуживания детерминированного потока требований. II.– Изв. АН СССР. Техн. кибернет., 1976, № 5.
5. *Шор Н. З.* Методы минимизации дефференцируемых функций и их приложения. К.: Наукова думка, 1979.
6. *Шор Н. З., Шабашов Л. П.* О решении минимаксных задач методом обобщенного градиентного спуска с растяжением пространства.– Кибернетика, 1972, № 1.

Киев

Поступила в редакцию
5.XI.1979