Модел. и анализ информ. систем. Т. **21**, № **4** (2014) 132–147

УДК 519.854.2

Эвристические алгоритмы для задачи целочисленного сбалансирования трехмерной матрицы с ограничениями второго рода

Смирнов А. В.

Ярославский государственный университет им. П. Г. Демидова
150000 Россия, г. Ярославль, ул. Советская, 14

e-mail: alexander_sm@mail.ru

получена 3 июня 2014

Ключевые слова: целочисленное сбалансирование, трехмерные матрицы, ограничения второго рода, кратные сети, кратные потоки, обобщенный алгоритм пометок, послойный алгоритм, матричный алгоритм

Рассматривается задача целочисленного сбалансирования с ограничениями второго рода. В вещественной трехмерной матрице элементы внутренней части (все три индекса больше нуля) просуммированы по каждому направлению и сечению матрицы, а также найдена общая сумма. Данные суммы размещаются в элементах матрицы, у которых один или несколько индексов равны нулю (в соответствии с направлениями суммирования). Ищется целочисленная матрица той же структуры, получаемая из исходной заменой элементов внутренней части на округления до целого сверху или целого снизу. При этом суммирующие элементы должны отклоняться от исходных менее чем на 2, а элемент с тремя нулевыми индексами получается по обычным правилам округления.

В статье разрабатываются эвристические алгоритмы решения задачи. Послойный алгоритм получается как обобщение соответствующего алгоритма для задачи с ограничениями первого рода. Предлагается новый матричный алгоритм, состоящий из трех частей: поиск базовой матрицы, поиск максимальной матрицы и коррекция матрицы. На всех шагах циклически проводятся изменения целочисленной матрицы, затрагивающие от 1 до 3 элементов внутренней части. Определяется модифицированный матричный алгоритм, направленный на более равномерное заполнение внутренней части целочисленной матрицы.

Также в статье оценивается сложность всех трех алгоритмов и приводится сравнительный анализ матричных алгоритмов на основе результатов вычислительных экспериментов.

1. Постановка задачи

Различные задачи целочисленного сбалансирования возникают в сфере управления, экономики, финансов. В частности, подобная задача ставится при планировании железнодорожных грузоперевозок. Имеется матричный план по отправке вагонов, который группируется по некоторым показателям (например, направление, тип вагона, владелец вагона и т. п.). Данный план составляется на месяц и естественно является целочисленным. Однако вагоны необходимо отправлять ежедневно. При делении на количество дней в месяце план перестает быть целочисленным. Поэтому возникает проблема такого округления основных параметров, чтобы суммирующие показатели не выходили за определенные рамки. Данный план может быть представлен в виде k-мерной матрицы, где k – это число показателей, по которым ведется суммирование.

В работах [1] – [3] была рассмотрена задача целочисленного сбалансирования трехмерной матрицы. Имеется трехмерная вещественная матрица A с неотрицательными элементами a_{ijp} ($i \in \overline{0,n}$, $j \in \overline{0,m}$, $p \in \overline{0,t}$), для которых выполнены условия баланса:

каждый элемент с некоторыми нулевыми индексами равен сумме всех элементов, для которых ненулевые индексы оставлены неизменными, а нулевые индексы заменены всеми возможными ненулевыми значениями диапазонов соответствующих индексов.

Требуется так округлить элементы матрицы до целых значений сверху или снизу (элемент a_{000} округляется до ближайшего целого), чтобы остались неизменными условия баланса. Данную задачу мы в дальнейшем будем называть *задачей целочисленного сбалансирования трехмерной матрицы с ограничениями первого рода.*

Условия округления, возникающие в этой задаче, фактически означают, что элементы итоговой матрицы отклоняются от элементов исходной матрицы менее чем на 1. Если же допустить, чтобы суммирующие элементы итоговой матрицы могли отклоняться от соответствующих элементов исходной матрицы (кроме элемента d_{000}) менее чем на 2, то мы получим постановку *задачи целочисленного сбалансирования трехмерной матрицы с ограничениями второго рода.* Так же, как и для первой задачи, можно без ограничения общности считать, что элементы a_{ijp} ($i > 0$, $j > 0$, $p > 0$) исходной матрицы находятся в интервале $[0, 1)$. Данная задача рассматривалась в работах [4] – [5]. Формально ее постановка выглядит следующим образом.

Имеется трехмерная вещественная матрица A с неотрицательными элементами a_{ijp} ($i \in \overline{0,n}$, $j \in \overline{0,m}$, $p \in \overline{0,t}$), для которых выполнены условия баланса:

$$a_{000} = \sum_{i=1}^{n}\sum_{j=1}^{m}\sum_{p=1}^{t} a_{ijp}; \quad a_{i00} = \sum_{j=1}^{m}\sum_{p=1}^{t} a_{ijp} \quad (i \in \overline{1,n}); \quad a_{0j0} = \sum_{i=1}^{n}\sum_{p=1}^{t} a_{ijp} \quad (j \in \overline{1,m});$$

$$a_{00p} = \sum_{i=1}^{n}\sum_{j=1}^{m} a_{ijp} \quad (p \in \overline{1,t}); \quad a_{ij0} = \sum_{p=1}^{t} a_{ijp} \quad (i \in \overline{1,n},\ j \in \overline{1,m});$$

$$a_{i0p} = \sum_{j=1}^{m} a_{ijp} \quad (i \in \overline{1,n},\ p \in \overline{1,t}); \quad a_{0jp} = \sum_{i=1}^{n} a_{ijp} \quad (j \in \overline{1,m},\ p \in \overline{1,t}).$$

Ищется целочисленная сбалансированная матрица D той же размерности, для которой выполнены условия:

$$\sum_{i=1}^{n}\sum_{j=1}^{m}\sum_{p=1}^{t} d_{ijp} = \lfloor a_{000} + 0.5 \rfloor;$$

$$\max\{0,\ \lfloor a_{i00} \rfloor - 1\} \le \sum_{j=1}^{m}\sum_{p=1}^{t} d_{ijp} \le \lceil a_{i00} \rceil + 1 \quad (i \in \overline{1,n});$$

$$\max\{0,\ \lfloor a_{0j0} \rfloor - 1\} \le \sum_{i=1}^{n}\sum_{p=1}^{t} d_{ijp} \le \lceil a_{0j0} \rceil + 1 \quad (j \in \overline{1,m});$$

$$\max\{0,\ \lfloor a_{00p} \rfloor - 1\} \le \sum_{i=1}^{n}\sum_{j=1}^{m} d_{ijp} \le \lceil a_{00p} \rceil + 1 \quad (p \in \overline{1,t});$$

$$\max\{0,\ \lfloor a_{ij0} \rfloor - 1\} \le \sum_{p=1}^{t} d_{ijp} \le \lceil a_{ij0} \rceil + 1 \quad (i \in \overline{1,n},\ j \in \overline{1,m});$$

$$\max\{0,\ \lfloor a_{i0p} \rfloor - 1\} \le \sum_{j=1}^{m} d_{ijp} \le \lceil a_{i0p} \rceil + 1 \quad (i \in \overline{1,n},\ p \in \overline{1,t});$$

$$\max\{0,\ \lfloor a_{0jp} \rfloor - 1\} \le \sum_{i=1}^{n} d_{ijp} \le \lceil a_{0jp} \rceil + 1 \quad (j \in \overline{1,m},\ p \in \overline{1,t});$$

$$\lfloor a_{ijp} \rfloor \le d_{ijp} \le \lceil a_{ijp} \rceil \quad (i \in \overline{1,n},\ j \in \overline{1,m},\ p \in \overline{1,t})$$

и сохраняются условия баланса.

2. Кратные сети и точный алгоритм решения

Известно, что задача целочисленного сбалансирования с ограничениями первого рода является NP-полной (см. [1]). Задача с ограничениями второго рода предположительно также является NP-полной. В связи с этим актуальным является вопрос построения достаточно эффективных алгоритмов целочисленного сбалансирования. В данной работе мы рассмотрим некоторые эвристические методы для задачи с ограничениями второго рода. Однако сначала кратко опишем идею построения точного (экспоненциального) алгоритма для этой задачи.

В работах [4] – [5] для решения задачи сбалансирования с ограничениями второго рода предлагалось сводить ее к решению задачи о наибольшем потоке в кратной сети кратности 2 (похожее сведение выполнялось ранее и для задачи с ограничениями первого рода, см. [2]). Напомним, что в качестве *кратной сети* произвольной натуральной кратности k рассматривается ориентированный мультиграф $G(X,U)$, между вершинами которого могут быть дуги одного из 3 видов:

1) *обычная дуга* u^o с пропускной способностью $c(u^o)$, поток по которой не связан с потоком по другим дугам; множество обычных дуг обозначим через U^o;

2) *кратная дуга* u^k между двумя вершинами, которая состоит из k дуг одной ориентации с одинаковой пропускной способностью $c(u^k)$ и одинаковым потоком по каждой из них; множество кратных дуг обозначим через U^k;

3) *связанная дуга* u между двумя вершинами, которая связана с еще $k-1$ дугой, имеющей одинаковый один из концов; множество связанных дуг, выходящих из одной вершины или входящих в одну вершину, будем называть *мультидугой* u^m; пропускная способность всех связанных дуг одной мультидуги одинакова; поток по каждой связанной дуге из мультидуги одинаков; множество мультидуг обозначим через U^m.

Множество выходящих из вершины дуг может быть либо только кратными дугами, либо только одной мультидугой (k связанных дуг), либо только обычными дугами. Из источника x_0 сети выходят только кратные дуги, а в сток z сети входит только одна мультидуга. Если из вершины выходят связанные дуги мультидуги, то в нее обязательно входит кратная дуга. Если в вершину входит мультидуга, то из нее может выходить только кратная дуга. Определенный таким образом мультиграф $G(X,U)$ с целочисленными пропускными способностями дуг назовем *кратной (транспортной) сетью.*

Кратным потоком по сети называется целочисленная функция, определенная на множестве дуг $U = U^o \cup U^k \cup U^m$, для которой выполнены условия неотрицательности, ограниченности (пропускными способностями дуг) и неразрывности потока (в каждой вершине). *Величиной кратного потока* называется сумма φ_z входящего потока для стока z, равная сумме выходящего из источника потока. В силу того, что поток по каждой обычной дуге и по каждой связанной дуге каждой кратной и мультидуги должен быть целочислен, величина φ_z должна быть кратна k. Как и обычно, обозначим через $c(u)$ *пропускную способность* дуги u, а через $f(u)$ – *поток* на ней.

Построим модель сведения к кратной сети кратности 2 для задачи сбалансирования с ограничениями второго рода. Каждому элементу a_{ijp} матрицы A соответствует вершина сети x_{ijp}. Каждой вершине x_{ijp}, где более чем 1 индекс имеет нулевые значения, соответствует дополнительная вершина x'_{ijp}, а также вершинам z_k $(k=1,2)$ соответствуют дополнительные вершины z'_k. Пропускные способности кратных дуг ($i \in \overline{1,n},\ j \in \overline{1,m},\ p \in \overline{1,t}$):

$$c(x_{000}, x_{i00}) = 2\max\{0,\ \lfloor a_{i00}\rfloor - 1\};\ c(x_{000}, x'_{000}) = 2\left(\lfloor a_{000} + 0.5\rfloor - \sum_{i=1}^{n} c(x_{000}, x_{i00})\right);$$

$$c(x'_{000}, x_{i00}) = 2(\lceil a_{i00}\rceil + 1 - c(x_{000}, x_{i00}));\ c(x_{i00}, x_{ij0}) = 2\max\{0,\ \lfloor a_{ij0}\rfloor - 1\};$$

$$c(x_{i00}, x'_{i00}) = 2\left(\lceil a_{i00}\rceil + 1 - \sum_{j=1}^{m} c(x_{i00}, x_{ij0})\right);\ c(x'_{i00}, x_{ij0}) = 2(\lceil a_{ij0}\rceil + 1 - c(x_{i00}, x_{ij0}));$$

$$c(x_{ij0}, x_{ijp}) = 2\lceil a_{ijp}\rceil.$$

Пропускные способности связанных дуг для всех мультидуг из вершин x_{ijp} равны $\lceil a_{ijp}\rceil$, а пропускные способности связанных дуг, входящих в вершину z : $c(z_k, z) = \lfloor a_{000} + 0.5\rfloor$ $(k=1,2)$.

Пропускные способности обычных дуг ($i \in \overline{1,n},\ j \in \overline{1,m},\ p \in \overline{1,t}$):

$$c(x_{0jp}, x_{0j0}) = \max\{0,\ \lfloor a_{0jp}\rfloor - 1\};\ c(x_{0jp}, x'_{0j0}) = \lceil a_{0jp}\rceil + 1 - c(x_{0jp}, x_{0j0});$$

$$c(x'_{0j0}, x_{0j0}) = \lceil a_{0j0} \rceil + 1 - \sum_{p=1}^{t} c(x_{0jp}, x_{0j0});\ c(x_{0j0}, z_1) = \max\{0,\ \lfloor a_{0j0} \rfloor - 1\};$$

$$c(x_{0j0}, z'_1) = \lceil a_{0j0} \rceil + 1 - c(x_{0j0}, z_1);\ c(z'_1, z_1) = \lfloor a_{000} + 0.5 \rfloor - \sum_{j=1}^{m} c(x_{0j0}, z_1);$$

$$c(x_{i0p}, x_{00p}) = \max\{0,\ \lfloor a_{i0p} \rfloor - 1\};\ c(x_{i0p}, x'_{00p}) = \lceil a_{i0p} \rceil + 1 - c(x_{i0p}, x_{00p});$$

$$c(x'_{00p}, x_{00p}) = \lceil a_{00p} \rceil + 1 - \sum_{i=1}^{n} c(x_{i0p}, x_{00p});\ c(x_{00p}, z_2) = \max\{0,\ \lfloor a_{00p} \rfloor - 1\};$$

$$c(x_{00p}, z'_2) = \lceil a_{00p} \rceil + 1 - c(x_{00p}, z_2);\ c(z'_2, z_2) = \lfloor a_{000} + 0.5 \rfloor - \sum_{p=1}^{t} c(x_{00p}, z_2).$$

Множество дуг (обычных, а также одной из двух связанных дуг каждой кратной и мультидуги), образующих все возможные обычные пути из x_{000} в z_1, назовем *частью* G_1 кратной сети целочисленного сбалансирования. Аналогичное множество дуг, образующих все возможные обычные пути из x_{000} в z_2, назовем *частью* G_2 кратной сети целочисленного сбалансирования.

Рассмотрим соотношения, которые мы будем называть *условиями разрешимости*:

$$\begin{gathered} f(x_{i00}, x_{ij0}) + f(x'_{i00}, x_{ij0}) \geq c(x_{i00}, x_{ij0})\ (i \in \overline{1,n},\ j \in \overline{1,m}); \\ f(x_{0jp}, x_{0j0}) + f(x_{0jp}, x'_{0j0}) \geq c(x_{0jp}, x_{0j0})\ (j \in \overline{1,m},\ p \in \overline{1,t}); \\ f(x_{i0p}, x_{00p}) + f(x_{i0p}, x'_{00p}) \geq c(x_{i0p}, x_{00p})\ (i \in \overline{1,n},\ p \in \overline{1,t}). \end{gathered} \tag{1}$$

В дальнейшем кратные и обычные дуги, инцидентные двум основным вершинам (без штриха), мы будем называть *основными*, а остальные кратные и обычные дуги мы будем называть *дополнительными*.

Теорема 1. Для того, чтобы задача целочисленного сбалансирования с ограничениями второго рода имела решение, необходимо и достаточно, чтобы в соответствующей ей кратной сети существовал максимальный кратный поток φ величины $\varphi_z = 2\lfloor a_{000} + 0.5 \rfloor$, для которого выполняются условия разрешимости (1).

Справедливость теоремы 1 следует непосредственно из правил построения сети и условия неразрывности потока в каждой вершине. Отметим также, что решение задачи о максимальном кратном потоке указанного вида будет индуцировать решение задачи сбалансирования с ограничениями второго рода. При этом d_{ijp} полагается равным половине величины потока, проходящего через вершину x_{ijp}, если она является концом кратной дуги; величине потока, проходящего через вершину x_{ijp}, если она является концом обычной дуги.

Напомним также несколько определений, касающихся кратных сетей целочисленного сбалансирования кратности 2, которые понадобятся нам в дальнейшем (подробнее см. в [2]).

Объединение мультидуги u_z^m, идущей из вершин z_1, z_2 в z и двух путей μ_r ($r = 1, 2$), каждый из которых является ориентированным путем в соответствующей части G_r из вершины x_{000} в вершину z_r, назовем *обобщенным путем*, если каждый из путей μ_r проходит через одну и ту же вершину x_{ijp} с ненулевыми индексами $i > 0$, $j > 0$, $p > 0$.

Кратный поток назовем *полным*, если любой обобщенный путь из x_{000} в z имеет дугу u (обычную, кратную или мультидугу), поток по которой равен ее пропускной способности $f(u) = c(u)$.

Проекция C_i $(i = 1, 2)$ подграфа C на часть сети G_i – это часть подграфа C, образованная его вершинами и дугами, принадлежащими G_i.

Так как части G_i $(i = 1, 2)$ сети G представляют собой обычные транспортные сети с источником x_{000} и стоком z_i, то будем называть некоторый путь из x_{000} в z_i *путем прорыва* в части G_i, если $f(u) < c(u)$ на прямых дугах и $f(u) > 0$ на обратных дугах этого пути.

Пусть в кратной сети определен некоторый поток φ величины $\varphi_z \geq 0$. *Кратным циклом* в сети $G(X, U)$, где X – множество вершин, U – множество дуг (обычных, кратных или мультидуг), назовем такой подграф $C(X', U')$, $X' \subseteq X$, $U' \subseteq U$, для которого:

1) проекции C_1 и C_2 на части G_1 и G_2 соответственно есть объединение некоторых циклов, причем дуги, поток по которым ненулевой, могут проходиться в обратном направлении;

2) проекции C_1 и C_2 согласованы (одинаковы) на общей части подграфов G_1 и G_2;

3) C_1 представим в виде $C_1 = \cup\{C_1^j\}$, где C_1^j – некоторые циклы и $C_1^j \not\subseteq C_1^k$ $\forall j \neq k$; при этом для любой дуги u из G_1 выполняется неравенство

$$0 \leq f(u) + a^+(u) - a^-(u) \leq c(u),$$

где $a^+(u)$ – это число циклов C_1^j, в которых дуга u проходится в прямом направлении, а $a^-(u)$ – это число циклов C_1^j, в которых дуга u проходится в обратном направлении. Такое же условие должно выполняться и для C_2.

Обобщенным путем прорыва в сети $G(X, U)$ для некоторого кратного потока φ назовем такой подграф $S(X', U')$, $X' \subseteq X$, $U' \subseteq U$, для которого:

1) каждая из проекций S_1 и S_2 на части G_1 и G_2 соответственно есть объединение ровно одного пути прорыва из x_{000} в z и некоторых циклов, причем дуги, поток по которым ненулевой, могут проходиться в обратном направлении;

2) проекции S_1 и S_2 согласованы (одинаковы) на общей части подграфов G_1 и G_2;

3) S_1 представим в виде $S_1 = \mu_1 \cup \{C_1^j\}$, где μ_1 – путь прорыва, C_1^j – некоторые циклы и $C_1^j \not\subseteq C_1^k$ $\forall j \neq k$; при этом для любой дуги u из G_1 выполняется неравенство

$$0 \leq f(u) + a^+(u) - a^-(u) \leq c(u),$$

где $a^+(u)$ – это число элементов множества $\mu_1 \cup \{C_1^j\}$, в которых дуга u проходится в прямом направлении, а $a^-(u)$ – это число элементов множества $\mu_1 \cup \{C_1^j\}$, в которых дуга u проходится в обратном направлении. Такое же условие должно выполняться и для S_2;

4) S не содержит кратного цикла.

В работе [4] показано, что решение задачи целочисленного сбалансирования трехмерной матрицы с ограничениями второго рода может быть найдено при помощи алгоритма поиска максимального кратного потока, удовлетворяющего условиям разрешимости (1), состоящего из трех этапов:

1. *Этап построения полного потока.* На данном этапе увеличения потока можно добиваться, находя все возможные обобщенные пути прорыва из x_{000} в z через вершины x_{ijp} ($i \in \overline{1,n},\ j \in \overline{1,m},\ p \in \overline{1,t}$) без обратных дуг и увеличивая поток по ним. В итоге будет получен полный поток.

2. *Этап увеличения потока.* Если полученный полный поток не является максимальным, то увеличиваем поток при помощи обобщенного алгоритма пометок (см. [6]) до тех пор, пока поток не станет максимальным. Напомним, что идея обобщенного алгоритма пометок состоит в следующем: в проекциях G_1 и G_2 поочередно строятся пути прорыва μ_1 и μ_2 (возможно, в объединении с некоторыми циклами) до тех пор, пока пути в обеих проекциях не станут согласованными, либо же не останется вариантов для продолжения построения пути. В первом случае объединение μ_1 и μ_2 с мультидугой с концом в z даст обобщенный путь прорыва, во втором случае производится откат до «точки ветвления», после чего выполнение алгоритма возобновляется. Если точкой ветвления оказывается x_{000} и при этом x_{000} не помечена, то задача целочисленного сбалансирования не имеет решения.

3. *Этап коррекции потока.* Если максимальный поток имеет величину $2\lfloor a_{000} + 0.5\rfloor$, но не удовлетворяет условиям разрешимости (1), то выполняем коррекцию потока при помощи модификации обобщенного алгоритма пометок (см. [2]), пока не будет получено выполнение условий или не будет установлена невозможность такой коррекции.

В дальнейшем при построении эвристических методов решения мы будем использовать разбиение алгоритма на этапы, подобные указанным выше.

Следует также отметить, что в работе [4] был построен алгоритм полиномиальной сложности для частного случая $n = 2$, $m = 2$ или $t = 2$. Данный алгоритм сводит задачу целочисленного сбалансирования трехмерной матрицы с ограничениями второго рода к двум задачам целочисленного сбалансирования двумерной матрицы (см. [7] – [8]), которые в свою очередь могут быть сведены к задаче поиска наибольшего потока в обычной сети (см. [9]).

3. Послойный алгоритм

Перейдем теперь к построению эвристических алгоритмов целочисленного сбалансирования. Первый подход к данной проблеме был осуществлен в работе [10], в которой был предложен *послойный алгоритм* для задачи целочисленного сбалансирования трехмерной матрицы с ограничениями первого рода. Оказывается, что данный алгоритм легко переносится и на задачу с ограничениями второго рода.

Введем обозначения для следующих множеств дуг в кратной сети целочисленного сбалансирования:

$$P_1 = \{(x_{i00}, x_{ij0}),\ (x_{i00}, x'_{i00}),\ (x'_{i00}, x_{ij0}),\ i \in \overline{1,n},\ j \in \overline{1,m}\};$$

$$P_2 = \{(x_{0jp}, x_{0j0}),\ (x_{0jp}, x'_{0j0}),\ (x'_{0j0}, x_{0j0}),\ j \in \overline{1,m},\ p \in \overline{1,t}\};$$

$$P_3 = \{(x_{i0p}, x_{00p}),\ (x_{i0p}, x'_{00p}),\ (x'_{00p}, x_{00p}),\ i \in \overline{1,n},\ p \in \overline{1,t}\}.$$

Алгоритм 1 *(послойный).* Для k от 1 до n последовательно выполним следующие шаги.

1. *Построение слоя.*

Включаем в k-й слой все вершины исходной сети вида x_{ijp} и x'_{ijp} для которых $i = 0$ или $i = k$. Включаем в слой также вершины z_1, z_2, z. При наличии включаем вершины z'_1, z'_2.

Включаем в слой все дуги исходной сети, соединяющие включенные ранее вершины. При этом пропускную способность каждой такой дуги (a, b) полагаем равной $c(a, b) - f(a, b)$, где $c(a, b)$ – это пропускная способность дуги в сети целочисленного сбалансирования, а $f(a, b)$ – поток на ней.

2. *Получение полного потока в слое.*

1) Увеличение потока с ограничениями на пропускную способность дуг:

а) временно считая нулевой пропускную способность дополнительных дуг в частях P_1, P_2, P_3, увеличиваем поток, пока возможно, находя обобщенные пути прорыва из x_{000} в z, все дуги в которых проходятся в прямом направлении.

Повторяем процедуру для следующих множеств дуг и в таком порядке:

б) P_2, P_3; в) P_1, P_2; г) P_1, P_3; д) P_3; е) P_1; ж)P_2.

2) Увеличение потока без ограничений на пропускную способность дуг.

3. *Получение максимального потока в слое.*

Если полный поток не является максимальным, то будем увеличивать его, находя обобщенные пути прорыва с помощью алгоритма красных и черных пометок (см. [10]). Идея алгоритма красных и черных пометок такова: сначала находятся пути прорыва μ_1, μ_2 в проекциях G_1 и G_2 соответственно, пометки этих путей объявляются черными. Если эти пути не являются согласованными на общей части, то непомеченные вершины x_{0jp} и x_{k0p} ($j > 0$, $p > 0$), соответствующие вершинам x_{kjp} путей μ_1, μ_2, получают красную пометку. После этого осуществляется поиск согласованного на общей части пути от вершин с красными пометками до вершин с черными пометками.

Получаемые таким образом обобщенные пути прорыва будут иметь простую структуру: ни одна дуга не будет проходиться в том или ином направлении дважды, кроме того, как правило, в таком пути будет не более 3 вершин вида x_{kjp} с ненулевыми индексами.

Если в результате применения алгоритма красных и черных пометок не будет найдено максимального потока в слое либо величина максимального потока будет меньше $\lfloor a_{k00} \rfloor - 1$, то послойный алгоритм неприменим к данной задаче.

4. *Увеличение потока в сети сбалансирования.*

Увеличиваем поток на каждой дуге сети целочисленного сбалансирования на соответствующую величину потока в k-м слое.

Если в результате выполнения шагов 1–4 для всех слоев величина потока в сети целочисленного сбалансирования будет меньше $2\lfloor a_{000} + 0.5 \rfloor$ или не будут выполняться условия разрешимости (1), то послойный алгоритм неприменим к данной задаче. В противном случае поток в сети индуцирует решение задачи целочисленного сбалансирования трехмерной матрицы с ограничениями второго рода.

Послойный алгоритм является наиболее быстрым из представленных в данной статье. Действительно, количество вершин в слое – $O(mt)$; трудоемкость шагов 1, 2, 4 линейно зависит от этого количества. Трудоемкость алгоритма красных и черных пометок линейно зависит от количества дуг в слое, которое также определяется как $O(mt)$. Шаги 1, 2, 4 применяются для каждого из n слоев, а алгоритм красных

и черных пометок – не более $\lfloor a_{000} \rfloor$ раз. Следовательно, трудоемкость послойного алгоритма составляет $O((n + a_{000})mt)$.

Однако существенным недостатком послойного алгоритма является то, что он практически не учитывает условий разрешимости (1) – ограничения, связанные с основными дугами, фигурирующими в данных соотношениях, используются только на этапе поиска полного потока; этап коррекции потока отсутствует. Также недостатком является невозможность изменить на k-м шаге поток в предшествующих слоях. В связи с этим послойный алгоритм оказывается неприменим к большинству примеров уже для небольших размерностей матрицы (например, при $n = m = t = 3$ и $a_{ijp} = \frac{1}{3}$ для всех $i > 0$, $j > 0$, $p > 0$ послойный алгоритм не находит решения). Поэтому возникает необходимость разработать более эффективные алгоритмы. Тем не менее, некоторые идеи из послойного алгоритма будут использованы нами при построении других методов.

4. Матричный алгоритм

Будем теперь строить эвристический алгоритм полиномиальной сложности, состоящий из этапов, подобных тем, которые используются в точном алгоритме. В целях сокращения количества вычислительныъх операций будем формулировать алгоритм сразу для матрицы, проводя при этом аналогии с соответствующей потоковой задачей. Сначала введем ряд определений.

Базовой матрицей назовем такую целочисленную трехмерную матрицу с элементами внутренней части из множества $\{0, 1\}$, в которой выполнены правые неравенства во всех условиях баланса, но замена любого из нулевых элементов внутренней части на 1 приводит к нарушению какого-либо из указанных условий.

Максимальной матрицей назовем такую целочисленную трехмерную матрицу, для которой элемент с нулевыми индексами равен $\lfloor a_{000} + 0.5 \rfloor$, выполнены условия баланса для элементов с одним или тремя ненулевыми индексами, а для элементов с двумя ненулевыми индексами выполнены правые неравенства из условий баланса.

Введем также константы

$$c_0 = \lfloor a_{000} + 0.5 \rfloor - \sum_{i=1}^{n} \max\{0, \lfloor a_{i00} \rfloor - 1\},$$

$$c_1 = \lfloor a_{000} + 0.5 \rfloor - \sum_{j=1}^{m} \max\{0, \lfloor a_{0j0} \rfloor - 1\},$$

$$c_2 = \lfloor a_{000} + 0.5 \rfloor - \sum_{p=1}^{t} \max\{0, \lfloor a_{00p} \rfloor - 1\}$$

и переменные f_0, f_1, f_2. Данные константы и переменные мы будем использовать для обеспечения выполнения условий баланса для элементов с одним ненулевым индексом. По сути, эти константы и переменные эквивалентны пропускной способности и потоку на дугах (x_{000}, x'_{000}), (z'_1, z_1) и (z'_2, z_2) кратной сети целочисленного сбалансирования.

Правило пересчета f_0. Если значение d_{ijp} ($i > 0$, $j > 0$, $p > 0$) меняется с 0 на 1 и изначально $d_{i00} \geq \lfloor a_{i00} \rfloor - 1$, то f_0 увеличивается на 1. Если значение d_{ijp} меняется с 1 на 0 и изначально $d_{i00} \geq \lfloor a_{i00} \rfloor$, то f_0 уменьшается на 1. Во всех остальных случаях f_0 не меняется.

Аналогичным образом (с точностью до индексов) определяются правила пересчета f_1 и f_2.

Перейдем теперь к алгоритму. На всех шагах алгоритма мы будем использовать сформулированные выше правила пересчета f_s ($s \in \overline{0,2}$).

Алгоритм 2 *(матричный).* Полагаем $d_{ijp} = 0$ для всех возможных значений индексов, $f_0 = f_1 = f_2 = 0$. Будем искать решение задачи в 3 этапа.

1. *Поиск базовой матрицы* (мы будем искать такую базовую матрицу, для которой выполнены условия $f_s \leq c_s$ ($s \in \overline{0,2}$), поэтому данный этап соответствует этапу поиска полного потока в точном алгоритме).

Для всех ненулевых значений a_{ijp} от a_{111} до a_{nmt} меняем значение d_{ijp} на 1 и пересчитываем суммирующие элементы, если при этом не будут нарушены правые неравенства в условиях баланса и все $f_s \leq c_s$.

Если $d_{000} = \lfloor a_{000} + 0.5 \rfloor$, то переходим на шаг 3.

2. *Поиск максимальной матрицы* (данный этап соответствует этапу поиска максимального потока в точном алгоритме, однако для полиномиальности алгоритма мы ограничиваемся поиском вариантов изменения матрицы, содержащих не более 3 элементов d_{ijp} с ненулевыми индексами).

Рассмотрим набор ненулевых индексов (i_1, j_1, p_1) такой, что $a_{i_1 j_1 p_1} > 0$ и $d_{i_1 j_1 p_1} = 0$. Значение $d_{i_1 j_1 p_1}$ меняем на 1 и пересчитываем суммирующие элементы, если при этом не будут нарушены правые неравенства в условиях баланса и все $f_s \leq c_s$.

Иначе рассматриваем тройку попарно различных наборов ненулевых индексов $\{(i_1, j_1, p_1), (i_2, j_2, p_2), (i_3, j_3, p_3)\}$ таких, что $a_{i_q j_q p_q} > 0$ ($q \in \overline{1,3}$), $d_{i_1 j_1 p_1} = d_{i_3 j_3 p_3} = 0$, $d_{i_2 j_2 p_2} = 1$. Меняем значения элементов на $d_{i_1 j_1 p_1} = d_{i_3 j_3 p_3} = 1$, $d_{i_2 j_2 p_2} = 0$ и пересчитываем суммирующие элементы, если при этом не будут нарушены правые неравенства в условиях баланса и все $f_s \leq c_s$.

Иначе переходим к следующей тройке. Если все тройки рассмотрены, переходим к следующему набору индексов (i_1, j_1, p_1).

Повторяем шаг 2 до тех пор, пока не станет $d_{000} = \lfloor a_{000} + 0.5 \rfloor$, либо на шаге 2 будет невозможно увеличить d_{000}. В первом случае переходим к следующему шагу; во втором случае матричный алгоритм неприменим к данной задаче.

3. *Коррекция матрицы* (данный этап соответствует этапу коррекции потока в точном алгоритме, однако мы ограничиваемся поиском вариантов изменения матрицы, содержащих 2 элемента d_{ijp} с ненулевыми индексами).

Пусть для некоторого набора ненулевых индексов (i_1, j_1) выполнено $d_{i_1 j_1 0} < \lfloor a_{i_1 j_1 0} \rfloor - 1$.

Рассмотрим такую пару наборов индексов $\{(i_1, j_1, p_1), (i_2, j_2, p_2)\}$, что $a_{i_q j_q p_q} > 0$ ($q \in \overline{1,2}$), $d_{i_1 j_1 p_1} = 0$, $d_{i_2 j_2 p_2} = 1$, кроме того, $i_1 \neq i_2$ или $j_1 \neq j_2$. Меняем значения элементов на $d_{i_1 j_1 p_1} = 1$, $d_{i_2 j_2 p_2} = 0$ и пересчитываем суммирующие элементы, если при этом не будут нарушены правые неравенства в условиях баланса и все $f_s \leq c_s$, а также не будут нарушены левые неравенства в тех условиях баланса, которые были выполнены до применения указанных изменений.

Иначе переходим к следующей паре наборов индексов. Если все пары рассмот-

рены, но по-прежнему выполнено $d_{i_1 j_1 0} < \lfloor a_{i_1 j_1 0} \rfloor - 1$, то матричный алгоритм к данной задаче неприменим.

Процедуру повторяем для всех таких наборов индексов (i_1, j_1).

Аналогичные (с точностью до индексов) действия осуществляем для наборов индексов (j_1, p_1), (i_1, p_1).

Оценим сложность матричного алгоритма. Очевидно, что первый шаг алгоритма линейно зависит от количества элементов внутренней части матрицы, а значит, его трудоемкость – $O(nmt)$. На втором шаге алгоритма осуществляется просмотр не более чем $\lfloor a_{000} \rfloor C^2_{nmt}$ троек элементов на каждом проходе цикла, причем цикл выполняется не более $\lfloor a_{000} \rfloor$ раз, следовательно, трудоемкость этапа – $O((a_{000} nmt)^2)$. На третьем шаге осуществляется просмотр не более чем $\lfloor a_{000} + 0.5 \rfloor (nmt - \lfloor a_{000} + 0.5 \rfloor)$ пар элементов, причем соответствующая процедура будет запущена не более $\lfloor a_{000} + 0.5 \rfloor$ раз, следовательно, трудоемкость этапа – $O((a_{000})^2 (nmt - a_{000}))$. В итоге сложность матричного алгоритма оценивается как $O((a_{000} nmt)^2)$.

5. Модификация матричного алгоритма

Построенный в предыдущем разделе матричный алгоритм является, безусловно, более эффективным, нежели послойный алгоритм. Однако в матричном алгоритме присутстсвует недостаток, связанный с неравномерным распределением единиц в целочисленной базовой матрице. Как следствие, это увеличивает количество примеров, для которых алгоритм оказывается неприменим (результаты соответствующих вычислительных экспериментов будут рассмотрены в следующем разделе). Указанную проблему можно частично решить за счет синтеза идей, использованных в послойном и матричном алгоритмах.

При построении модифицированного матричного алгоритма будем в различных сочетаниях использовать следующие ограничения:

$$d_{ij0} < \max\{0,\ \lfloor a_{ij0} \rfloor - 1\} \quad (i \in \overline{1,n},\ j \in \overline{1,m}); \tag{2}$$

$$d_{0jp} < \max\{0,\ \lfloor a_{0jp} \rfloor - 1\} \quad (j \in \overline{1,m},\ p \in \overline{1,t}); \tag{3}$$

$$d_{i0p} < \max\{0,\ \lfloor a_{i0p} \rfloor - 1\} \quad (i \in \overline{1,n},\ p \in \overline{1,t}). \tag{4}$$

Алгоритм 3 *(матричный, модифицированный).* Полагаем $d_{ijp} = 0$ для всех возможных значений индексов, $f_0 = f_1 = f_2 = 0$. Будем искать решение задачи в 3 этапа.

1. *Поиск базовой матрицы.*

1) Поиск базовой матрицы с использованием дополнительных ограничений:

а) для всех ненулевых значений a_{ijp} от a_{111} до a_{nmt}: если для данного набора (i, j, p) выполнены условия (2), (3), (4), то меняем значение d_{ijp} на 1 и пересчитываем суммирующие элементы, если при этом не будут нарушены правые неравенства в условиях баланса и все $f_s \le c_s$.

Если $d_{000} = \lfloor a_{000} + 0.5 \rfloor$, то переходим на шаг 3.

Повторяем процедуру для следующих наборов ограничений и в таком порядке:

б) (3), (4); в) (2), (3); г) (2), (4); д) (4); е) (2); ж) (3).

2) Повторяем процедуру поиска без использования дополнительных ограничений.

2. *Поиск максимальной матрицы.* Аналогично матричному алгоритму.

3. *Коррекция матрицы.* Аналогично матричному алгоритму.

Очевидно, что трудоемкость модифицированного матричного алгоритма зависит прежде всего от трудоемкости второго этапа (так же, как и в алгоритме 2). Следовательно, порядок сложности будет тот же – $O((a_{000}nmt)^2)$.

6. Сравнительный анализ матричных алгоритмов

Для выполнения сравнительного анализа двух матричных алгоритмов (алгоритм 2 и алгоритм 3) была написана программа на языке $C\#$. Следует отметить, что для матричных алгоритмов имеет существенное значение направление просмотра элементов при поиске решения: в одном и том же примере матричный алгоритм с одним направлением просмотра найдет решение, с другим – нет. Более того, реализации с чередованием направления просмотра будут эффективней, поскольку так можно добиться более равномерного распределения значений в целочисленной матрице. В указанной программной реализации использовался следующий подход к организации просмотра элементов:

– этап 1, матричный алгоритм: от a_{111} к a_{nmt};

– этап 1, модифицированный матричный алгоритм: шаг 1) – от a_{111} к a_{nmt}; шаг 2) – от a_{nmt} к a_{111};

– этап 2: (i_1, j_1, p_1) – от (n, m, t) к $(1, 1, 1)$; (i_2, j_2, p_2) – от $(1, 1, 1)$ к (n, m, t); (i_3, j_3, p_3) – от $(1, 1, 1)$ к (n, m, t);

– этап 3: (i_1, j_1, p_1) – от $(1, 1, 1)$ к (n, m, t); (i_2, j_2, p_2) – от $(1, 1, 1)$ к (n, m, t).

Вычислительные эксперименты проводились с использованием ПК со следующими характеристиками: процессор – Intel Core i5-2500K (4326.7MHz, 4 ядра), RAM – 16 Gb. Испытания проводились сериями по 200 экспериментов, внутренние части исходных трехмерных матриц получались при помощи датчиков следующих распределений (датчики строились по стандартным алгоритмам, см., например, [11]):

1) равномерное;

2) экспоненциальное с функцией распределения $f(x) = e^{-x},\ x \geq 0$;

3) нормальное $N(0, 1)$;

4) нормальное $N(0.5, 0.25)$;

5) нормальное $N(0.75, 0.1)$.

Эксперименты формировались из тестов двух типов:

1) *0.1-тесты* – матрицы с десятичными элементами из интервала $[0, 1)$, получаемыми при помощи одного из указанных распределений и содержащими один знак после запятой;

2) *0.5-тесты* – матрицы, состоящие из элементов 0 и 0.5, получаемых при помощи одного из распределений.

Для кубических матриц было проведено 12000 тестов для размерностей внутренней части матрицы от $3 \times 3 \times 3$ до $8 \times 8 \times 8$ (по 200 тестов для каждой тройки {размерность, распределение, тип теста}).

Среднее время выполнения тестов (в секундах):

Размерность	**0.1-тесты**		**0.5-тесты**	
	Алгоритм 2	**Алгоритм 3**	**Алгоритм 2**	**Алгоритм 3**
$3 \times 3 \times 3$	0.00007	0.00007	0.00002	0.00004
$4 \times 4 \times 4$	0.00063	0.00072	0.00013	0.00017
$5 \times 5 \times 5$	0.01921	0.01086	0.00194	0.00211
$6 \times 6 \times 6$	0.34848	0.12211	0.04607	0.01611
$7 \times 7 \times 7$	2.68833	0.47303	0.40331	0.09682
$8 \times 8 \times 8$	11.48674	2.02440	1.65551	0.29394

Статистика успехов/неудач для 0.1-тестов («решение» – решение найдено; «выход» – решение не найдено):

Размерность	**Алгоритм 2**		**Алгоритм 3**		**Выход в обоих алгоритмах**
	Решение	**Выход**	**Решение**	**Выход**	
$3 \times 3 \times 3$	1000	0	1000	0	0
$4 \times 4 \times 4$	1000	0	1000	0	0
$5 \times 5 \times 5$	863	137	992	8	1
$6 \times 6 \times 6$	533	467	857	143	66
$7 \times 7 \times 7$	81	919	829	171	158
$8 \times 8 \times 8$	5	995	721	279	277

Статистика успехов/неудач для 0.5-тестов:

Размерность	**Алгоритм 2**		**Алгоритм 3**		**Выход в обоих алгоритмах**
	Решение	**Выход**	**Решение**	**Выход**	
$3 \times 3 \times 3$	1000	0	1000	0	0
$4 \times 4 \times 4$	999	1	1000	0	0
$5 \times 5 \times 5$	981	19	999	1	0
$6 \times 6 \times 6$	905	95	997	3	0
$7 \times 7 \times 7$	548	452	967	33	16
$8 \times 8 \times 8$	158	842	933	67	58

Анализируя результаты экспериментов, следует отметить следующее:

1) как правило, модифицированный алгоритм выполняется быстрее матричного; это, очевидно, связано с тем, что модифицированный алгоритм обеспечивает более равномерное заполнение внутренней части целочисленной матрицы, а следовательно, этап поиска максимальной матрицы и этап коррекции выполняются меньшее число раз;

2) по той же причине количество успешно решенных примеров больше для алгоритма 3;

3) 0.5-тесты выполняются быстрее 0.1-тестов для одинаковых размерностей матрицы, и количество успехов в них выше для обоих алгоритмов; это обстоятельство связано с тем, что в 0.5-тестах содержится большое количество нулей во внутренней части исходной матрицы, а время выполнения каждого алгоритма и вероятность успеха зависят прежде всего от количества ненулевых значений;

4) наибольшее среднее время выполнения теста обоими алгоритмами было показано на тестах, генерируемых с помощью датчика нормального распределения

$N(0.5, 0.25)$; в соответствующих матрицах большинство элементов не равно 0 и близко (для 0.1-тестов) или равно (для 0.5-тестов) значению 0.5;

5) наименьшее количество успешно решенных примеров для обоих алгоритмов получено на тестах, генерируемых с помощью датчиков нормального распределения $N(0.5, 0.25)$ и $N(0.75, 0.1)$;

6) единственный пример размерности $4\times4\times4$, для которого алгоритм 2 не нашел решения, – это матрица с элементами $a_{ijp} = 0.5$ при всех $i > 0$, $j > 0$, $p > 0$;

7) для всех шести размерностей находились примеры, когда один алгоритм находит решение, а другой нет, и наоборот. Это приводит к идее использования комбинированного варианта на практике с целью повышения количества успешно решенных примеров. Наряду с комбинированием алгоритмов 2 и 3 можно предложить также комбинирование различных реализаций алгоритма 3 (отличающихся направлениями просмотра), либо более сложные варианты с поворотом матрицы или перестановкой сечений по i, j или p.

В целом же можно заключить, что для рассмотренных размерностей данные алгоритмы являются достаточно эффективными. Действительно, решение не было найдено ни одним из алгоритмов в 8.37 % случаев для 0.1-тестов и всего лишь в 1.23 % случаев для 0.5-тестов.

Примечательными являются также тесты, в которых одна из размерностей равна 3. Если для алгоритма 2 статистика примерно соответствует кубическим тестам с аналогичным количеством элементов внутренней части матрицы (отличие лишь в том, что скорость убывания количества успешных тестов и скорость роста времени выполнения несколько ниже), то для алгоритма 3 ситуация принципиально иная. Статистика для 20000 тестов указанного вида для алгоритма 3 следующая:

Размерность	**0.1-тесты**			**0.5-тесты**		
	Решение	**Выход**	**Время**	**Решение**	**Выход**	**Время**
$3 \times 8 \times 8$	997	3	0.03067	1000	0	0.00552
$3 \times 9 \times 9$	997	3	0.06523	998	2	0.01726
$3 \times 10 \times 10$	998	2	0.12602	999	1	0.03306
$3 \times 11 \times 11$	998	2	0.23477	998	2	0.06131
$3 \times 12 \times 12$	997	3	0.44986	991	9	0.10748
$3 \times 13 \times 13$	995	5	0.72894	995	5	0.18516
$3 \times 14 \times 14$	995	5	1.15870	998	2	0.28602
$3 \times 15 \times 15$	975	25	1.87494	990	10	0.49505
$3 \times 16 \times 16$	975	25	3.93859	998	2	0.42212
$3 \times 17 \times 17$	958	42	4.67662	998	2	1.08196

Таким образом, в случае $n = 3$ алгоритм 3 оказывается эффективен для достаточно больших размерностей внутренней части матрицы. В проведенных тестах решение не было найдено в 1.15 % случаев для 0.1-тестов и в 0.35 % случаев для 0.5-тестов, причем 82 из 115 нерешенных примеров для 0.1-тестов и 11 из 35 нерешенных примеров для 0.5-тестов пришлись на матрицы, полученные с помощью датчика нормального распределения $N(0.75, 0.1)$.

Список литературы

1. Рублев В.С., Смирнов А.В. NP-полнота задачи целочисленного сбалансирования трехмерной матрицы // ДАН. 2010. Т. 435, №3. С. 314–316 (English transl.: Roublev V.S., Smirnov A.V. NP-Completeness of the Integer Balancing Problem for a Three-Dimensional Matrix // Doklady Mathematics. 2010. Vol. 82, №3. P. 912–914).

2. Рублев В.С., Смирнов А.В. Задача целочисленного сбалансирования трехмерной матрицы и алгоритмы ее решения // Моделирование и анализ информационных систем. 2010. Т. 17, №2. С. 72–98 (Roublev V.S., Smirnov A.V. The Problem of Integer-Valued Balancing of a Three-Dimensional Matrix and Algorithms of Its Solution // MAIS. 2010. V. 17, №2. P. 72–98 [in Russian]).

3. Смирнов А.В. Задача целочисленного сбалансирования трехмерной матрицы и сетевая модель // Моделирование и анализ информационных систем. 2009. Т. 16, №3. С. 70–76 (Smirnov A.V. The Problem of Integer-valued Balancing of a Three-dimensional Matrix and Network Model // MAIS. 2010. V. 16, №3. P. 70–76 [in Russian]).

4. Смирнов А.В. Некоторые классы разрешимости задачи целочисленного сбалансирования трехмерной матрицы с ограничениями второго рода // Моделирование и анализ информационных систем. 2013. Т. 20, №2. С. 54–69 (Smirnov A.V. Some Solvability Classes for The Problem of Integer Balancing of a Three-dimensional Matrix with Constraints of Second Type // MAIS. 2013. V. 20, №2. P. 54–69 [in Russian]).

5. Смирнов А.В. Задача целочисленного сбалансирования трехмерной матрицы с ограничениями второго рода // Моделирование и анализ информационных систем: Труды международной научной конференции, посвященной 35-летию математического факультета и 25-летию факультета информатики и вычислительной техники Ярославского государственного университета им. П.Г. Демидова. Ярославль, 2012. С. 164–167 [Smirnov A.V. Zadacha tselochislennogo sbalansirovaniya tryokhmernoy matritsy s ogranicheniyami vtorogo roda // Modelirovanie i analiz informatsionnykh sistem: Trudy mezhdunarodnoy nauchnoy konferentsii, posvyashchyonnoy 35-letiyu matematicheskogo fakulteta i 25-letiyu fakulteta informatiki i vychislitelnoy tekhniki Yaroslavskogo gosudarstvennogo universiteta im. P. G. Demidova. Yaroslavl, 2012. S. 164–167 (in Russian)].

6. Рублев В. С., Смирнов А. В. Потоки в кратных сетях // Ярославский педагогический вестник. 2011. Т. 3, №2. С. 60–68 (Rublev V.S., Smirnov A.V. Flows in Multiple Networks // Yaroslavsky Pedagogichesky Vestnik. 2011. V. 3, №2. P. 60–68 [in Russian]).

7. Кондаков А.С., Рублев В.С. Задача сбалансирования матрицы плана // Доклады Одесского семинара по дискретной математике. Одесса: Астропринт, 2005. Вып. 2. С. 24–26 [Kondakov A.S., Roublev V.S. Zadacha sbalansirovaniya matritsy plana // Doklady Odesskogo seminara po diskretnoy matematike. Odessa: Astroprint, 2005. №2. S. 24–26 (in Russian)].

8. Коршунова Н.М., Рублев В.С. Задача целочисленного сбалансирования матрицы // Современные проблемы математики и информатики. Ярославль, 2000. Вып. 3. С. 145–150 [Korshunova N.M., Roublev V.S. Zadacha tselochislennogo sbalansirovaniya matritsy // Sovremennye problemy matematiki i informatiki. Yaroslavl, 2000. №3. S. 145–150 (in Russian)].

9. Форд Л.Р., Фалкерсон Д.Р. Потоки в сетях. М.: Мир, 1966. 276 с. (Ford L.R., Fulkerson D.R. Flows in Networks. Princeton University Press, 1962. 194 p.).

10. Рублев В.С., Смирнов А.В. Задача оптимального округления плана валютных счетов // Кибернетика и высокие технологии XXI века. Воронеж: НПФ «Саквоее», 2008. Т. 1. С. 112–123 [Roublev V.S., Smirnov A.V. Zadacha optimalnogo okrugleniya plana valyutnykh schetov // Kibernetika i vysokie tekhnologii XXI veka. Voronezh: NPF "SAKVOEE", 2008. V. 1. S. 112–123 (in Russian)].

11. Михайлов Г.А., Войтишек А.В. Численное статистическое моделирование. Методы Монте-Карло. М.: Академия, 2006. 368 с. (Mikhaylov G.A., Voytishek A.V. Chislennoe statisticheskoe modelirovanie. Metody Monte-Karlo. M.: Akademiya, 2006. 368 s. (in Russian)].

Heuristic Algorithms for The Problem of Integer Balancing of a Three-dimensional Matrix with Constraints of Second Type

Smirnov A. V.

P.G. Demidov Yaroslavl State University, Sovetskaya str., 14, Yaroslavl, 150000, Russia

Keywords: integer balancing, three-dimensional matrices, constraints of second type, multiple networks, multiple flows, generalized labeling algorithm, layering algorithm, matrix algorithm

The problem of integer balancing of a three-dimensional matrix with constraints of second type is studied. The elements of the inner part (all three indices are greater than zero) of the three-dimensional matrix are summed in each direction and each section of the matrix; the total sum is also found. These sums are placed into the elements where one or more indices are equal to zero (according to the summing directions). The problem is to find an integer matrix of the same structure, which can be produced from the initial one by replacing the elements of the inner part with the largest previous or the smallest following integer. At the same time, variations of the sums of elements from those in the initial matrix should be less than 2 and the element with three zero indices should be produced with standard rules of rounding-off. Heuristic algorithms for this problem are suggested in the article: layering algorithm being got as generalization of a similar algorithm for the problem with constraints of first type and a new matrix algorithm. The last one consists of three parts: search for the base matrix, search for the maximal matrix and matrix correcting. Each of them is a cyclic change of the integer matrix using from 1 to 3 elements from the inner part. A modification of the matrix algorithm is suggested. The algorithm is directed to more uniform filling of the inner part of the integer matrix. Also, the complexity of all three algorithms is estimated in the article. The comparative analysis of matrix algorithms based on the results of computing experiments is adduced.

Сведения об авторе:
Смирнов Александр Валерьевич,
Ярославский государственный университет им. П.Г. Демидова,
канд. физ.-мат. наук, доцент кафедры теоретической информатики