УДК 519.854.2:004.023

А.Н. ТРОФИМЧУК, В.А. ВАСЯНИН, В.Н. КУЗЬМЕНКО

О СЛОЖНОСТИ ОДНОЙ ЗАДАЧИ ОПТИМИЗАЦИИ УПАКОВОК

Аннотация. Рассмотрена задача оптимизации упаковок элементов квадратной матрицы, заданных целыми положительными числами, в блоки фиксированного размера. Предложена постановка задачи и исследована трудоемкость полного перебора ее решений. Доказано, что задача является NP-полной, путем полиномиального сведения к ней NP-полной целочисленной задачи о многопродуктовом потоке минимальной стоимости.

Ключевые слова: оптимизация упаковок, целочисленный многопродуктовый поток, трудоемкость полного перебора, полиномиальная сводимость, NP-полные задачи.

ВВЕДЕНИЕ

Рассмотренная далее задача оптимизации упаковок, в которой элементы строк и столбцов квадратной матрицы, заданные целыми положительными числами, определенным образом упаковываются в блоки фиксированного размера, возникает при формализации и оптимизации процессов сортировки и упаковки мелкопартионных корреспонденций (грузов или сообщений) в транспортные блоки (физические или виртуальные контейнеры) в транспортных сетях или в сетях передачи данных [1, 2]. Суть прикладной задачи заключается в нахождении рациональной схемы объединения исходящих из узлов сети мелкопартионных потоков однородных грузов (сообщений) с различными адресами назначения и упаковки их в транспортные блоки (контейнеры) заданного размера в целях минимизации количества последних для транспортировки мелкопартионных корреспонденций во всей сети при определенных условиях ее функционирования.

В данной работе рассматривается постановка «чистой» задачи упаковок без функций затрат, которые можно ввести для оценки стоимости процессов обработки и транспортировки мелкопартионных корреспонденций в реальных приложениях. При этом данная прикладная задача сводится к задаче над элементами квадратной матрицы. Исследуется переборная сложность задачи и доказано, что она является NP-полной. Приводятся данные экспериментальных исследований, показывающие зависимость результатов решения задачи от границ изменения значений входных данных и схем перебора вариантов решения.

ПОСТАНОВКА ЗАДАЧИ

Задана матрица $A = ||a_{ij}||_{n\times n}$ с целыми положительными элементами вне диагонали и с нулевой диагональю. Над элементами матрицы можно выполнять некоторую операцию объединения, соблюдая определенные ограничения. Для выполнения отдельной операции выбирается элемент $a_{ij} > 0$ и непустое множество непересекающихся индексов $\{k_1, k_2, \ldots, k_m\}$, отличных от i и j ($\bigcap_{m=1}^{n-2} k_m = \varnothing$, $n \geq 3$). В результате элементы $a_{ik_1}, a_{k_1k_2}, \ldots, a_{k_mj}$ увеличиваются на величину a_{ij} и элемент a_{ij} обнуляется. Каждый преобразованный $a_{ik_1}, a_{k_1k_2}, \ldots$ $\ldots, a_{k_mj}$ или непреобразованный элемент a_{ij} (когда $\{k_1, k_2, \ldots, k_m\} = \varnothing$) размещается («упаковывается») в целое количество блоков размера $\omega \in Z^+$. При изменении матрицы изменяется суммарное количество блоков, необходимых для размещения всех элементов. Требуется найти такую последовательность операций объединения, чтобы в результате получить матрицу с минимальным

числом блоков. Содержательно исходная матрица A задает потребности в транспортировке, а операция объединения означает, что если есть прямой поток a_{ij}, то его можно направить через транзитные узлы $k_1, k_2, \dots, k_m$, соответственно изменив элементы $a_{ik_1}, a_{k_1k_2}, \dots, a_{k_mj}$, a_{ij}.

Приведем формальную постановку задачи, разделив исходное состояние матрицы и измененное. Имеется матрица целочисленных переменных $X = ||x_{ij}||_{n\times n}$, начальные значения которой совпадают со значениями матрицы A. Требуется найти минимум функции

$$\sum_{i=1}^{n}\sum_{j=1}^{n}\lceil x_{ij}/\omega\rceil, x_{ij} = \begin{cases} a_{ij} + \sum\limits_{rs} a^*_{rs}, & \text{если } a_{ij} \text{ непосредственно не объединялся} \\ & \text{с другими элементами,} \\ 0 & \text{в противном случае,} \end{cases} \quad (1)$$

$$i, j = \overline{1, n},$$

где $\lceil x\rceil$ — наименьшее целое, большее или равное x, $\{a^*_{rs}\}$ — множество элементов (может быть пустым), объединенных с a_{ij}. Условия баланса имеют вид

$$\sum_{j=1}^{n} x_{ij} - \sum_{j=1}^{n} a_{ij} = \sum_{j=1}^{n} x_{ji} - \sum_{j=1}^{n} a_{ji}, \ i = \overline{1, n}. \quad (2)$$

Ограничения на величину значений x_{ij} определятся так:

$$x_{ij} \le \left\lceil \frac{a_{ij}}{\omega} \right\rceil \omega, \ i, j = \overline{1, n}. \quad (3)$$

Оценим трудоемкость решения сформулированной задачи. При построении алгоритма нужно решить, как выбирать элементы x_{ij} и индексы $\{k_1, k_2, \dots, k_m\}$ для выполнения операции объединения. При решении задачи (1)–(3) в матрице X имеется $n(n-1)$ вариантов выбора первого элемента x_{ij}. Ясно, что после выбора этого элемента и какого-либо множества индексов $\{k_1, k_2, \dots, k_m\}$ существует огромное число последующих вариантов выбора x_{ij} и индексов. Учитывая, что объединение элементов x_{ij} может выполняться многократно, число вариантов полного перебора решения задачи (1)–(3), которое будет расти экспоненциально, непросто выразить в виде формулы от размерности матрицы X. Если сформулировать (1)–(3) как задачу перечисления, т.е. ответить на вопрос, сколько у нее имеется решений, то, скорее всего, она принадлежит к классу KP-полных задач [3, стр. 210]. При решении индивидуальной задачи существенную роль играют границы изменения значений элементов a_{ij} по сравнению со значением ω. Так, если $a_{ij} \ge \omega \ \ \forall i, j = \overline{1, n}$, то очевидно, что верхняя граница минимума равна $\sum_{i=1}^{n}\sum_{j=1}^{n}\lceil a_{ij}/\omega\rceil$ и определить это можно за время $O(n^2)$, поскольку операции объединения элементов могут только увеличить значение функции цели. Если выполняется $\sum_{j=1}^{n} a_{ij} \le \omega \ \ \forall i = \overline{1, n}$ и $\sum_{i=1}^{n} a_{ij} \le \omega \ \ \forall j = \overline{1, n}$, то в этом крайнем случае существует простой алгоритм, не учитывающий ограничения (3), со сложностью $O(n^3)$ для нахождения нижней границы минимума функции $\sum_{i=1}^{n}\sum_{j=1}^{n}\lceil x_{ij}/\omega\rceil = 2n-2$ с минимальным значением транзитных объединенных элементов $y = \sum_{i=1}^{n} y_i$, $y_i = \sum_{j=1}^{n} x_{ij} - \sum_{j=1}^{n} a_{ij}$ (см. приведенный далее алгоритм unconditional packing).

Лемма 1. Максимальное число элементов, преобразуемых с помощью операции объединения, не может превышать значения $n^2 - 3n + 2$.

Доказательство. Число элементов в матрице равно $n^2 - n$. Поскольку любую матрицу с ненулевыми элементами (кроме диагональных) с помощью операции объединения можно привести к виду, в котором она будет содержать ровно $2n-2$ элемента, число преобразуемых элементов определяется как $n^2 - n - (2n-2) = n^2 - 3n + 2$.

NP-ПОЛНОТА ЗАДАЧИ

Отметим, что задача (1)–(3), сформулированная как задача распознавания: существует ли последовательность объединений такая, что значение функции (1) не превышает $B \in Z^+$, принадлежит к классу NP, так как для нее всегда найдется сертификат с длиной, ограниченной полиномом $n^2 - 3n + 2$ (лемма 1). Покажем, что NP-полную целочисленную задачу о многопродуктовом потоке минимальной стоимости (minimum-cost integer multicommodity flow problem, MC IMCF) можно полиномиально свести к задаче (1)–(3). В работе [4] NP-полнота задачи MC IMCF даже для двухпродуктового целочисленного потока доказана путем сведения к ней задачи 3-выполнимости (3-SAT). Под сводимостью здесь и далее подразумевается сводимость по Карпу для соответствующих задач распознавания [3].

Приведем постановку задачи MC IMCF в форме «узлы–дуги» (node–arc or conventional). Пусть $G(N, E)$ — сеть с множеством узлов N, $n = |N|$, и множеством ориентированных дуг $ij \in E$, $e = |E|$. Задано множество продуктов (commodity) P, $p = |P|$, каждый из которых характеризуется тройкой (s_m, t_m, a_m), $m = 1, 2, \ldots, p$, $a_m \in \mathrm{Z}^+$, где s_m, t_m и a_m — соответственно узлы отправления (origin) и назначения (destination), а также количество (demand) продукта $m \in P$, которое нужно доставить из узла s_m в узел t_m. При транспортировке продукта запрещено его разветвление (расщепление, дробление на части). Поскольку все продукты однородны, предполагается, что единица любого продукта при транспортировке по дуге использует единицу ее пропускной способности. Для каждой дуги сети $ij \in E$ заданы пропускная способность $w_{ij} \in \mathrm{Z}^+$ и стоимость $c_{ij}^m > 0$, $c_{ij}^m \in \mathrm{Z}^+$, транспортировки единицы продукта m. Введем булевы переменные x_{ij}^m, $ij \in E$, $m \in P$. Так как разветвление потоков запрещено, то $x_{ij}^m = 1$, если продукт m транспортируется по дуге $ij \in E$, и $x_{ij}^m = 0$ в противном случае.

Требуется найти

$$\min \sum_{m \in P} \sum_{ij \in E} c_{ij}^m a_m x_{ij}^m \tag{4}$$

при ограничениях

$$\sum_{j: ij \in E} x_{ij}^m - \sum_{j: ji \in E} x_{ji}^m = \begin{cases} 1 & \text{при } i = s_m, \\ 0 & \text{при } i \neq s_m \neq t_m, \\ -1 & \text{при } i = t_m \end{cases} \quad \forall m \in P, \ \forall i \in N; \tag{5}$$

$$\sum_{m \in P} a_m x_{ij}^m \leq w_{ij} \quad \forall ij \in E; \tag{6}$$

$$x_{ij}^m \in \{0, 1\} \quad \forall ij \in E, \ \forall m \in P. \tag{7}$$

Отметим, что для задачи MC IMCF известно много разновидностей постановок, например с матричной записью ограничений баланса (5) с использованием матрицы инциденций узлы–дуги сети [5]. Формулировки в виде «дуги–пути» (arc–path or column generation) приведены в [6, 7]. Все постановки задач MC IMCF при различных дополнительных ограничениях можно легко трансформировать одну в другую. В [6, 7] приведен также обширный библиографический обзор развития методов и алгоритмов решения многопродуктовых задач.

Теорема 1. Задача упаковок (1)–(3) является NP-полной.

Доказательство. Для сведения задачи (4)–(7) к задаче (1)–(3) рассмотрим индивидуальную задачу (4)–(7) на полной сети, когда индексы m всех продуктов заменены номерами узлов s_m и t_m, совпадающими с индексами дуг (т.е. для каждого продукта $st \in P$ имеется дуга $ij \in E$ и $s = i,\ t = j$).

Индивидуальная задача. Пусть $G(N, E)$ — полная сеть без петель с множеством узлов N, $n = |N|$, и множеством ориентированных дуг $ij \in E$, $e = |E| = n^2 - n$, т.е. $\forall\, ij \in E\ \exists\, ji \in E$. Каждая дуга ij имеет пропускную способность $w_{ij} \in \mathrm{Z}^+$. Задана матрица требований (demand) $A = ||a_{st}||_{n\times n}$ и число ω, a_{st}, $\omega \in Z^+$, $a_{st} = 0$ при $s = t$. Каждый продукт $st \in P$, $|P| = n^2 - n$, в количестве a_{st} должен быть доставлен из узла отправления s в узел назначения t. Пусть стоимость транспортировки единицы продукта $c_{ij}^{st} = 1\ \forall\, ij \in E$ и $\forall\, st \in P$. Переменные $x_{ij}^{st} = 1$, если продукт $st \in P$ транспортируется по дуге $ij \in E$, и $x_{ij}^{st} = 0$ в противном случае. Пусть $w_{ij} = \left\lceil \dfrac{a_{ij}}{\omega} \right\rceil \omega\ \forall\, ij \in E$. Обозначим $x_{ij} = \sum\limits_{st\in P} a_{st} x_{ij}^{st}\ \forall\, ij \in E$. Тогда индивидуальная задача записывается так

$$\min \sum_{ij\in E} \left\lceil \frac{x_{ij}}{\omega} \right\rceil \tag{8}$$

при ограничениях

$$\sum_{j:ij\in E} a_{st} x_{ij}^{st} - \sum_{j:ji\in E} a_{st} x_{ji}^{st} = \begin{cases} a_{st} & \text{при } i = s, \\ 0 & \text{при } i \neq s \neq t, \\ -a_{st} & \text{при } i = t, \end{cases} \qquad \forall\, st \in P,\ \forall\, i \in N; \tag{9}$$

$$x_{ij} \le \left\lceil \frac{a_{ij}}{\omega} \right\rceil \omega\ \ \forall\, ij \in E; \tag{10}$$

$$x_{ij}^{st} \in \{0,1\}\ \ \forall\, ij \in E,\ \forall\, st \in P. \tag{11}$$

Сложив для каждого i равенства (9) по всем продуктам $st \in P$, получим уравнения баланса для узлов i в виде

$$\sum_{j:\, ij\in E} \sum_{st\in P} a_{st} x_{ij}^{st} - \sum_{j:\, ji\in E} \sum_{st\in P} a_{st} x_{ji}^{st} = \sum_{t:\, it\in P} a_{it} - \sum_{s:\, si\in P} a_{si}$$

или

$$\sum_{j:\, ij\in E} x_{ij} - \sum_{t:\, it\in P} a_{it} = \sum_{j:\, ji\in E} x_{ji} - \sum_{s:\, si\in P} a_{si}\ \ \forall\, i \in N,$$

что идентично условиям (2).

Таблица 1

Постановка задачи	Количество	
	ограничений	переменных
Исходная (1)–(3)	$n + (n^2 - n)$	$(n^2 - n)$
В виде MC IMCF узлы–дуги (8)–(11)	$(n + 1)(n^2 - n)$	$(n^2 - n)^2$

Таким образом, задача (1)–(3) имеет решение тогда и только тогда, когда имеет решение построенная задача (8)–(11). Приведенное сведение можно выполнить за полиномиальное время $O(n^2)$, преобразовав любую сеть с заданными требованиями $A = ||a_{st}||_{n\times n}$ в полную с дугами, для которых $c_{ij}^{st} = 1$ и $w_{ij} = \left\lceil \frac{a_{ij}}{\omega} \right\rceil \omega$ для всех индексов *st* и *ij*. Пусть имеется любое (не обязательно оптимальное) решение x_{ij}^{st}, $st \in P$, $ij \in E$, задачи (8)–(11). Набор значений x_{ij}^{st} однозначно определяет $x_{ij} = \sum_{st\in P} a_{st} x_{ij}^{st} \ \forall\, ij \in E$ — значения переменных в (1)–(3). Поэтому очевидно, что решения задач (8)–(11) и (1)–(3) совпадают. Обратно. Пусть получено какое-либо решение x_{ij}, $i, j = \overline{1, n}$, задачи (1)–(3). По сохраненным цепочкам объединения элементов всегда можно определить значения x_{ij}^{st}, отличные от нуля. Таким образом, по решению одной задачи можно получить аналогичное решение другой задачи и наоборот. Очевидно, что задача (1)–(3) при формулировке ее в виде задачи MC IMCF на полной сети полиномиально сводится к задаче (8)–(11). В силу NP-полноты (8)–(11) задача (1)–(3) также является NP-полной.

Теорема доказана.

Сравнивая различные постановки задачи для полной сети с числом продуктов $p = n^2 - n$ и с числом дуг $e = n^2 - n$ по количеству ограничений и переменных (табл. 1), можно утверждать, что для разработки эвристических комбинаторных алгоритмов решения задачи лучшей является исходная формулировка, так как она содержит значительно меньше ограничений и переменных. Постановка (1)–(3) дает возможность конструировать эффективные (полиномиальные) алгоритмы локальной оптимизации (алгоритмы метода вектора спада, рандомизированные, метаалгоритмы, гибридные и пр.) без введения булевых переменных (т.е. не использовать в качестве базы только известные методы решения целочисленной задачи о многопродуктовом потоке минимальной стоимости [6, 7]).

ЧИСЛЕННЫЙ ЭКСПЕРИМЕНТ

Покажем на примере простых тестовых алгоритмов упаковок (unconditional packing и conditional packing), как влияют изменение границ входных данных и различные переборные схемы решения задачи на конечные результаты. Эти алгоритмы не предназначены для оптимального решения задачи (1)–(3) и тем более для решения практических задач упаковок с заданными функциями затрат на обработку и транспортировку мелкопартионных корреспонденций, в основе которых лежит решение задачи (1)–(3). Результаты работы этих алгоритмов только указывают, на что нужно обратить внимание при дальнейшей разработке рабочих алгоритмов для решения задач упаковок в реальных коммуникационных сетях.

Введем некоторую фиксированную операцию объединения элементов при решении задачи (1)–(3), которая формально записывается так: над любой трой-

кой элементов $x_{ik}, x_{kj}, x_{ij} \neq 0$, $k, i, j = \overline{1, n}$, $k \neq i \neq j$, матрицы X могут выполняться действия

$$x_{ik} \leftarrow x_{ik} + x_{ij},\ x_{kj} \leftarrow x_{kj} + x_{ij},\ c_{ij} \leftarrow k,\ y_k \leftarrow y_k + x_{ij},\ x_{ij} \leftarrow 0, \qquad (12)$$

где «←» — знак операции присваивания; k — индекс столбца и строки, через который выполняется объединение x_{ij}, причем x_{ij} должен объединяться с x_{ik} и x_{kj} только целиком; c_{ij} — элементы справочной матрицы объединений элементов $C = ||c_{ij}||_{n\times n}$; $Y = ||y_k||$, $k = \overline{1, n}$, — вектор сумм транзитных объединенных элементов, $y_k = \sum_{j=1}^{n} x_{kj} - \sum_{j=1}^{n} a_{kj}$. В начальном состоянии для всех i элементы $c_{ij} = i\ \forall j = \overline{1, n}$, а $Y = 0$. Из матрицы C на любом шаге можно определить с помощью алгоритмов, приведенных в [8], последовательность $\Omega_{ij} = \{(i, k_1),\ (k_1, k_2), \ldots, (k_m, j)\}$ с промежуточными индексами столбцов и строк $\{k_1, k_2, \ldots, k_m\}$, через которые выполнялось объединение элемента a_{ij}, а также найти множество других элементов $\{a^*_{rs}\}$, которые уже были объединены с a_{ij} на предыдущих итерациях, если $c_{ij} = i$.

Лемма 2. Если

$$\Delta = \left\lceil \frac{x_{ik}}{\omega} \right\rceil + \left\lceil \frac{x_{kj}}{\omega} \right\rceil + \left\lceil \frac{x_{ij}}{\omega} \right\rceil - \left(\left\lceil \frac{x_{ik} + x_{ij}}{\omega} \right\rceil + \left\lceil \frac{x_{kj} + x_{ij}}{\omega} \right\rceil \right), \qquad (13)$$

то ее значение не может превышать единицы.

Доказательство. В (13) возможны три случая: $\Delta < 0$, $\Delta = 0$ и $\Delta = 1$. Последний следует из того, что при условиях $\left\lceil \frac{x_{ik} + x_{ij}}{\omega} \right\rceil = \left\lceil \frac{x_{ik}}{\omega} \right\rceil$ и $\left\lceil \frac{x_{kj} + x_{ij}}{\omega} \right\rceil = \left\lceil \frac{x_{kj}}{\omega} \right\rceil$ разность в (13) не может превышать 1, так как при $x_{ij} \geq \omega$ и $\left\lceil \frac{x_{ij}}{\omega} \right\rceil \geq 1$ она всегда будет отрицательной.

Таким образом, за одну операцию объединения (12) при $\Delta = 1$ можно уменьшить значение функции (1) на единицу.

Рассмотрим алгоритмы unconditional packing и conditional packing и результаты их работы при изменении границ значений $\min \leq (x_{ij} = a_{ij}) \leq \max$ при фиксированном ω. В алгоритмах используются очевидное свойство $\sum_i \lceil x_i / \omega \rceil \geq \left\lceil (\sum_i x_i) / \omega \right\rceil$ и то, что любую матрицу X с помощью преобразований (12) можно для любых строки i и столбца j, $i = j$, привести к виду (в примере $i = j = 1$):

$$\begin{Vmatrix} 0 & \sum_i x_{i2} & \ldots & \sum_i x_{in} \\ \sum_j x_{2j} & 0 & 0 & 0 \\ \ldots & 0 & 0 & 0 \\ \sum_j x_{nj} & 0 & 0 & 0 \end{Vmatrix}.$$

Алгоритм unconditional packing имеет сложность $O(n^3)$, не проверяет выполнения ограничений (3) и значения Δ в (13). Алгоритм conditional packing проверяет значение Δ и при $\Delta \geq 0$ допускает ослабление ограничений (3). Сложность второго алгоритма больше и составляет $O(fs \cdot n^3)$, где fs — число входов в цикл по k.

Алгоритм unconditional packing

1. $OPT\_X \leftarrow A$; $OPT\_CS \leftarrow C$; $opt\_cont\_sum \leftarrow \sum_{i=1}^{n}\sum_{j=1}^{n}\lceil a_{ij}/\omega\rceil$; $OPT\_Y \leftarrow 0$; $opt\_y\_sum \leftarrow 0$.

2. Для $\{k \mid k = \overline{1,n}\}$ выполнить пп. 3–11.

3. $X \leftarrow A$; $CS \leftarrow C$; $Y \leftarrow 0$.

4. Для $\{i \mid i = \overline{1,n},\ i \neq k\}$ выполнить пп. 5–8.

5. Для $\{j \mid j = \overline{1,n},\ j \neq i \wedge j \neq k\}$ выполнить пп. 6–7.

6. $x_{ik} \leftarrow x_{ik} + x_{ij}$; $x_{kj} \leftarrow x_{kj} + x_{ij}$; $cs_{ij} \leftarrow k$; $y_k \leftarrow y_k + x_{ij}$; $x_{ij} \leftarrow 0$.

7. Перейти к п. 5. *** Конец цикла по j

8. Перейти к п. 4. *** Конец цикла по i

9. $cont\_sum \leftarrow \sum_{r=1}^{n}\sum_{s=1}^{n}\lceil x_{rs}/\omega\rceil$, $y\_sum \leftarrow \sum_{r=1}^{n} y_r$.

10. Если $cont\_sum < opt\_cont\_sum \vee cont\_sum = opt\_cont\_sum \wedge y\_sum < opt\_y\_sum$, то $OPT\_X \leftarrow X$; $OPT\_CS \leftarrow CS$; $OPT\_Y \leftarrow Y$; $opt\_cont\_sum = cont\_sum$; $opt\_y\_sum \leftarrow y\_sum$.

11. Перейти к п. 2. *** Конец цикла по k

12. Конец алгоритма.

Алгоритм conditional packing

1. $OPT\_X \leftarrow A$; $OPT\_CS \leftarrow C$; $opt\_cont\_sum \leftarrow \sum_{i=1}^{n}\sum_{j=1}^{n}\lceil a_{ij}/\omega\rceil$; $OPT\_Y \leftarrow 0$; $fs \leftarrow 0$; $fs\_cont\_sum \leftarrow opt\_cont\_sum$; $FS\_Y \leftarrow 0$; $fs\_y\_sum \leftarrow 0$.

2. Для $\{k \mid k = \overline{1,n}\}$ выполнить пп. 3–12.

3. $X \leftarrow A$; $CS \leftarrow C$; $Y \leftarrow FS\_Y$; $opt\_y\_sum \leftarrow fs\_y\_sum$.

4. Для $\{i \mid i = \overline{1,n},\ i \neq k\}$ выполнить пп. 5–9.

5. Для $\{j \mid j = \overline{1,n},\ j \neq i \wedge j \neq k \wedge x_{ik} \neq 0 \wedge x_{kj} \neq 0 \wedge x_{ij} \neq 0\}$ выполнить пп. 6–8.

6. $\Delta \leftarrow \left\lceil \frac{x_{ik}}{\omega} \right\rceil + \left\lceil \frac{x_{kj}}{\omega} \right\rceil + \left\lceil \frac{x_{ij}}{\omega} \right\rceil - \left(\left\lceil \frac{x_{ik}+x_{ij}}{\omega} \right\rceil + \left\lceil \frac{x_{kj}+x_{ij}}{\omega} \right\rceil \right)$.

7. Если $\Delta > 0$, то $x_{ik} \leftarrow x_{ik} + x_{ij}$; $x_{kj} \leftarrow x_{kj} + x_{ij}$; $cs_{ij} \leftarrow k$; $y_k \leftarrow y_k + x_{ij}$; $x_{ij} \leftarrow 0$. *** Возможно и $\Delta \geq 0$

8. Перейти к п. 5. *** Конец цикла по j

9. Перейти к п. 4. *** Конец цикла по i

10. $cont\_sum \leftarrow \sum_{r=1}^{n}\sum_{s=1}^{n}\lceil x_{rs}/\omega\rceil$, $y\_sum \leftarrow \sum_{r=1}^{n} y_r$.

11. Если $cont\_sum < opt\_cont\_sum \vee cont\_sum = opt\_cont\_sum \wedge y\_sum < opt\_y\_sum$, то $OPT\_X \leftarrow X$; $OPT\_CS \leftarrow CS$; $OPT\_Y \leftarrow Y$; $opt\_cont\_sum = cont\_sum$; $opt\_y\_sum \leftarrow y\_sum$.

12. Перейти к п. 2. *** Конец цикла по k

13. Если $opt\_cont\_sum < fs\_cont\_sum$, то $fs \leftarrow fs+1$; $fs\_cont\_sum \leftarrow opt\_cont\_sum$; $A \leftarrow OPT\_X$; $C \leftarrow OPT\_CS$; $FS\_Y \leftarrow OPT\_Y$; $fs\_y\_sum \leftarrow opt\_y\_sum$; перейти к п. 2.

14. Конец алгоритма.

В записи алгоритмов прописными буквами обозначены матрицы и векторы (например, OPT_X, OPT_Y), а строчными (например, opt_cont_sum, opt_y_sum) — простые переменные, знаком *** отмечены комментарии. В алгоритме unconditional packing Δ из (13) не вычисляется, в этом случае ограничения (3) не учитываются. В алгоритме conditional packing в п. 7 допускается $\Delta > 0$ и $\Delta \geq 0$, что соответствует строгому соблюдению ограничений (3) и их ослаблению.

Таблица 2

Интервалы значений a_{ij}, $i, j = \overline{1,n}$	Результаты работы алгоритмов unconditional packing и conditional packing					
	opt_cont_sum	*opt_y_sum*	K_{av}	N_{av}	fs	t_{av}, с
[1, 5]	9900	0	0,075	99	–	–
	1569	29057	0,936	1	1	0,17
	1570	29126	0,937	1	3	0,97
	2391	22477	0,519	23	30	6,46
[1, 10]	9900	0	0,138	99	–	–
	2793	53253	0,963	1	1	0,16
	2790	53278	0,942	2	3	0,97
	3552	32890	0,597	35	34	8,75
[1, 20]	9900	0	0,262	99	–	–
	5238	101619	0,982	1	1	0,17
	5219	100735	0,866	2	7	1,54
	4510	47325	0,819	45	59	17,02
[1, 40]	9900	0	0,512	99	–	–
	9900	0	0,512	99	1	0,16
	8039	92435	0,833	35	31	7,13
	6734	28156	0,854	67	75	29,36
[1, 60]	13232	0	0,550	99	–	–
	13232	0	0,550	99	1	0,17
	12069	149615	0,887	37	45	10,94
	9931	43588	0,868	65	98	37,47
[1, 80]	14869	0	0,632	99	–	–
	14869	0	0,632	99	1	0,16
	14057	119340	0,889	51	36	10,81
	12334	32342	0,872	73	93	39,91
[1, 100]	17827	0	0,657	99	–	–
	17827	0	0,657	99	1	0,16
	17049	122630	0,893	54	40	12,93
	15344	37448	0,873	74	93	40,34
[1, 120]	19791	0	0,701	99	–	–
	19791	0	0,701	99	1	0,16
	19160	102874	0,900	61	39	14,04
	17726	30478	0,881	78	92	42,46

При проведении эксперимента матрица A генерировалась датчиком псевдослучайных чисел для восьми интервалов. Результаты эксперимента для $n = 100$ и $\omega = 40$ приведены в табл. 2, в которой приняты следующие обозначения: *opt_cont_sum* — значение функции цели (1); *opt_y_sum* — минимальное значение $y = \sum_{k=1}^{n} y_k$; K_{av} — средний коэффициент загрузки блока; N_{av} — среднее число ненулевых элементов в строке матрицы X; f_s — число входов в цикл по k в алгоритме conditional packing; t_{av} — время решения задачи. Для каждого интервала данной таблицы результаты соответствуют работе алгоритмов: до оптимизации — первая строка; unconditional packing — вторая строка; conditional packing при $\Delta \geq 0$ и $\Delta > 0$ — третья и четвертая строки.

Значения $K_{av} = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{\delta_i}\sum_{j=1}^{n} x_{ij} \Big/ \left(\left\lceil \frac{x_{ij}}{\omega} \right\rceil \omega\right)$, $\delta_i = \sum_{j=1}^{n} \delta_{ij}$, и $N_{av} = \frac{1}{n}\sum_{i=1}^{n} \delta_i$, $i = \overline{1,n}$, вычислялись по результирующей матрице OPT_X, где $\delta_{ij} = 1$, если $x_{ij} \neq 0$, и $\delta_{ij} = 0$ в противном случае.

В табл. 3 приведены результаты работы алгоритма conditional packing при $\Delta \geq 0$, $\Delta > 0$ и перестановке индексов основного тройного цикла (пп. 2, 4, 5 алго-

Таблица 3

Последовательность индексов тройного цикла	Результаты работы алгоритма conditional packing					
	opt_cont_sum	opt_y_sum	K_{av}	N_{av}	fs	t_{av}, с
k, i, j	46481	0	0,514	199	–	–
	40729	1438516	0,856	60	65	106,63
	32220	434132	0,877	127	196	573,91
k, j, i	46481	0	0,514	199	–	–
	40729	1438516	0,856	60	65	105,89
	32220	434132	0,877	127	196	571,85
i, k, j	46481	0	0,514	199	–	–
	41463	1537721	0,934	64	192	339,69
	32779	501658	0,890	130	200	591,79
j, k, i	46481	0	0,514	199	–	–
	41431	1532645	0,921	64	191	338,27
	32798	497041	0,880	130	199	584,35
i, j, k	46481	0	0,514	199	–	–
	42599	1595685	0,941	80	169	332,05
	34729	670593	0,886	140	200	644,47
j, i, k	46481	0	0,514	199	–	–
	42645	1601494	0,919	80	175	362,27
	34692	661718	0,881	140	200	700,57

ритма) при $n = 200$, $\omega = 100$, $a_{ij} \in [1, 120]$, $i, j = \overline{1, n}$. В первой строке даны результаты работы агоритма conditional packing до оптимизации, во второй и третьей строках — соответственно при $\Delta \geq 0$ и $\Delta > 0$.

Из табл. 2 видно, что алгоритм unconditional packing может при малых границах изменения a_{ij} дать лучшие решения, чем алгоритм conditional packing. При больших значениях ω с помощью этого алгоритма можно получить и оптимальное решение $opt\_cont\_sum = 2n - 2 = 198$ с минимальным значением opt_y_sum. При увеличении границ изменения a_{ij} лучшие результаты показывает алгоритм conditional packing при $\Delta \geq 0$ и $\Delta > 0$.

Из табл. 3 можно видеть, что различные переборные схемы объединения элементов в алгоритме conditional packing дают разные сильно отличающиеся результаты.

Анализ результатов, приведенных в табл. 2 и 3, позволяет сделать следующие выводы:

— при малых значениях a_{ij}, $i, j = \overline{1, n}$ (интервалы [1, 5], [1, 10]) лучшие значения функции цели можно получить с помощью алгоритма unconditional packing, когда Δ не вычисляется и ограничения (3) не учитываются, а начиная с интервала [1, 40] этот алгоритм неприменим, так как не приводит к улучшению значения целевой функции;

— для интервалов [1, 5], [1, 10] неплохие результаты показывает алгоритм conditional packing при $\Delta \geq 0$, в этом случае сумма значений транзитных объединенных элементов (opt_y_sum) всегда намного больше, чем в случае $\Delta > 0$, что необходимо учитывать при разработке рабочих алгоритмов для реальных задач упаковки со сложной целевой функцией;

— безусловная и условная при $\Delta \geq 0$ стратегии могут расширять границы поиска локального минимума при случайном их использовании в алгоритме conditional packing и привести к лучшему результату по сравнению с непрерывным применением стратегии $\Delta > 0$;

— средний коэффициент загрузки блока K_{av} не может являться критерием поиска лучшего решения;

— данные эксперимента подтверждают сильную зависимость результатов решения задачи от выбора элементов x_{ij} и индексов k для выполнения операций объединения (12) и от границ изменения входных данных — значений n, a_{ij}, ω, а также необходимость поиска и разработки эффективных эвристических алгоритмов.

Все задачи решались на ПК с процессором Intel Core 2 Duo с тактовой частотой 2,66 ГГц и оперативной памятью 2 Гб.

ЗАКЛЮЧЕНИЕ

В работе доказана NP-полнота одной задачи оптимизации упаковок, которая используется при сортировке и упаковке мелкопартионных грузов в контейнеры в транспортных сетях или объединении сообщений в виртуальные контейнеры в сетях передачи данных. Приведены данные численного эксперимента с простыми тестовыми алгоритмами упаковок, которые показывают сильную зависимость результатов решения задачи от границ изменения значений входных данных и выбора схем перебора вариантов решения, а также указывают, на что нужно обратить внимание при разработке алгоритмов для решения задач обработки и транспортировки мелкопартионных корреспонденций в реальных коммуникационных сетях.

СПИСОК ЛИТЕРАТУРЫ

1. Трофимчук А.Н., Васянин В.А. Методика решения задачи оптимизации упаковок для управления перспективным развитием узлов коммуникационной сети // Кибернетика и системный анализ. — 2014. — № 4. — С. 88–99.
2. Трофимчук А.Н., Васянин В.А. Моделирование упаковки, распределения и маршрутизации мелкопартионных потоков в многопродуктовой сети // Проблемы управления и информатики. — 2015. — № 4. — С. 132–146.
3. Гэри М., Джонсон Д. Вычислительные машины и труднорешаемые задачи. — М.: Мир, 1982. — 416 с.
4. Even S., Itai A., Shamir A. On the complexity of timetable and multicommodity flow problems // SIAM J: Comput. — 1976. — **5.** — P. 691–703.
5. Ahuja R.K., Magnanti T.L., Orlin J.B. Network flows: theory, algorithms, and applications. — Upper Saddle River (New Jersey): Prentice-Hall, Inc., 1993. — 846 p.
6. Barnhart C., Hane C.A., Vange P.H. Using branch-and-price-and-cut to solve origin destination integer multicommodity flow problems // Operations Research. — 2000. — **48**, N 2. — P. 318–326.
7. Barnhart C., Krishnan N., Vange P.H. Multicommodity flow problems: In Encyclopedia of Optimization: Second Edition, C.A. Floudas and P.M. Pardalos (Eds.). — New York: Springer, 2009. — P. 2354–2362.
8. Васянин В.А. Справочная матрица слияния потоков в задачах оптимизации упаковок на многопродуктовых сетях // Системні дослідження та інформаційні технології. — 2014. — № 3. — С 42–49.

Поступила 27.04.2015