*УДК 519.8*

**М.С. Баркетов[1], Х. Копфер[2], Е. Пеш[3]**

# НИЖНИЕ ГРАНИЦЫ ЦЕЛЕВОЙ ФУНКЦИИ В ЗАДАЧЕ НАЗНАЧЕНИЯ ПОЕЗДОВ НА ВРЕМЕННЫЕ ИНТЕРВАЛЫ

*Рассматривается задача оптимизации выполнения множества операций по перемещению контейнеров на железнодорожном узле. Разрабатываются нижняя граница, основанная на релаксированной задаче линейного программирования, и нижняя граница, полученная с помощью техники лагранжевой релаксации.*

## Введение

В настоящей работе рассматривается задача оптимизации выполнения множества операций по перемещению контейнеров на железнодорожном узле, см. работы Бойсена и др. [1, 2]. Пусть дано множество $I=\{1, 2,\ldots,n\}$ поездов (где $n$ – это количество поездов), на каждом из которых находится определенное количество контейнеров. Число контейнеров на поезде $j$, которые должны быть перемещены на поезд $i$, равно $A_{ij}$. Множество поездов должно быть разбито на подмножества. Каждое подмножество может состоять не более чем из $G$ поездов, где $G$ – количество путей железнодорожного узла. Поезда каждого подмножества заезжают на железнодорожные пути узла одновременно и покидают железнодорожный узел одновременно. Дано $T$ временных интервалов $1, 2,\ldots,T$. Временной интервал соответствует номеру подмножества поездов. Подмножество, назначенное на первый временной интервал, обслуживается первым. После того как это первое подмножество покидает узел, обслуживается подмножество поездов, назначенное на второй временной интервал, и т. д. Без ограничения общности предполагается, что $n=GT$, иначе можно добавить поезда, не несущие и не получающие контейнеры. Поезд $i$ может быть назначен в подмножество между его наименьшим возможным временным интервалом $e_i$ и его самым поздним возможным временным интервалом $l_i$. Задача состоит в том, чтобы разбить множество всех поездов на $T$ подмножеств, по $G$ поездов каждое, и назначить эти подмножества на временные интервалы $1, 2, \ldots, T$ таким образом, чтобы каждый поезд был назначен на допустимый временной интервал. Перед формулировкой критерия оптимальности введем понятия повторных визитов и перемещений контейнеров через хранилище. Если есть поезда с контейнерами для поезда $i$ и эти поезда назначены на более поздние временные интервалы, чем временной интервал поезда $i$, тогда для поезда $i$ необходимо повторное посещение узла, чтобы забрать все необходимые контейнеры. Эти повторные посещения осуществляются после того, как все подмножества поездов посетили узел, потому что в этом случае ни одному поезду не нужно повторно посещать узел. Если два поезда назначены на один и тот же временной интервал, перемещение контейнеров между ними можно осуществлять напрямую. Если два поезда назначены на разные временные интервалы, необходимы перемещения контейнеров через хранилище. Ясно, что предпочтительнее перемещать контейнеры напрямую, а не через хранилище. В работе ставится задача поиска такого разбиения множества поездов на подмножества и назначения этих подмножеств поездов на временные интервалы, чтобы линейная комбинация количества повторных визитов поездов и перемещений контейнеров через хранилище была минимальна.

Хотя в литературе много внимания было уделено оптимизации железнодорожных перевозок в общем (см. Кордо и др. [3], Мачарис и Бонтеконинг [4], Бонтеконинг и др. [5] и Крэник и Ким [6]), статей по рассматриваемому новому типу железнодорожных узлов не так уже и много. В частности, известны только две работы, в которых исследуется рассматриваемая здесь задача: Бойсен и др. [1, 2]. В этих работах авторы доказывают, что рассматриваемая задача NP-трудна, и предлагают несколько эффективных точных и приближенных алгоритмов ее решения.

В данной статье предлагаются методы получения двух новых нижних границ оптимального значения целевой функции рассматриваемой задачи. Первая граница получается на основе релаксированной задачи линейного программирования. Другая граница получается в результате применения техники лагранжевой релаксации ограничения на размерность подмножества. Лагранжева релаксация – это мощный инструмент, который позволяет строить хорошие нижние границы. Идея лагранжевой релаксации заключается в релаксировании множества ограничений и добавлении их в целевую функцию с некоторыми коэффициентами. Лагранжева релаксация применялась с успехом ко многим задачам, см., например, работы Хугевина и Ван Де Вельде [7], Фишера [8], Ван Де Вельде [9], Фишетти и Тоса [10].

**1. Модели целочисленного математического программирования**

Сформулируем задачу целочисленного математического программирования:
*минимизировать*

$$f(x) = cx = \alpha_1 \sum_{i \in I} y_i + \alpha_2 \sum_{i \in I} \sum_{j \in L_i} z_{ij} A_{ij}, \tag{1}$$

*где* $x = (y_1, ..., y_n, x_{1,1}, ..., x_{nT}, z_{1,2}, z_{1,3}, ..., z_{2,1}, z_{2,3}, ..., z_{n1}, z_{n2}, ..., z_{n,n-1})$, а $L_i$ – *множество поездов, на которых расположены контейнеры для поезда* $i$,
*при условиях*

$$\sum_{t=e_i}^{l_i} x_{it} = 1 \text{ для } i \in I; \tag{2}$$

$$\sum_{i \in I} x_{it} \le G \text{ для } t,\ 1 \le t \le T; \tag{3}$$

$$\sum_{t=1}^{T} t x_{it} + M y_i \ge \sum_{t=1}^{T} t x_{jt} \text{ для } i \in I,\ \forall j \in L_i; \tag{4}$$

$$\left| \sum_{t=1}^{T} t x_{it} - \sum_{t=1}^{T} t x_{jt} \right| \le M z_{ij} \text{ для } i, j \in I,\ i \ne j; \tag{5}$$

$$x_{i,t}, y_i, z_{ij} \in \{0,1\},\ i, j \in I,\ i \ne j,\ 1 \le t \le T; \tag{6}$$

$$x_{it} = 0 \text{ для } i \in I,\ t > l_i \quad \text{или} \quad t < e_i. \tag{7}$$

В данной формулировке переменная $x_{it}$ равна 1, если поезд $i$ выбран во временной слот $t$, и равна 0 в противном случае. Переменная $y_i$ равна 1, если поезд $i$ повторно посещает узел, и равна 0 в противном случае. Переменная $z_{ij}$ равна 1, если поезда $i$ и $j$ назначены на разные временные интервалы, и равна 0 в противном случае. Число $M$ – это некоторое большое целое число; например, $M$ может быть равным $T$.

Ограничения (2) гарантируют, что каждый поезд назначен ровно на один допустимый временной интервал; (3) ограничивают количество поездов в подмножестве количеством имеющихся путей; (4) гарантируют правильные значения переменных $y_i$; (5) определяют значения переменных $z_{ij}$; (6) являются ограничениями на целочисленность; (7) гарантируют, что никакой поезд не будет назначен на недопустимый интервал.

Отметим, что формулировка (1)–(7) слабая, так как оптимальное значение релаксированной задачи линейного программирования, в которой условия целочисленности переменных заменяются условиями принадлежности их соответствующим отрезкам, равно нулю. Однако можно модифицировать формулировку (1)–(7) следующим образом:
*минимизировать*

$$f(x) = cx = \alpha_1 \sum_{i \in I} y_i + \alpha_2 \sum_{i \in I} \sum_{j \in L_i} z_{ij} A_{ij} \tag{8}$$

*при условиях*

$$\sum_{t=e_i}^{l_i} x_{it} = 1 \text{ для } i \in I; \tag{9}$$

$$\sum_{i\in I} x_{it} \le G \text{ для } t,\ 1 \le t \le T; \tag{10}$$

$$y_i \ge a_{ij} \text{ для } i \in I,\ j \in L_i; \tag{11}$$

$$z_{ij} = a_{ij} + a_{ji} \text{ для } i,\ j \in I,\ i \ne j; \tag{12}$$

$$\sum_{t=1}^{\tau} x_{jt} + a_{ij} \ge \sum_{t=1}^{\tau} x_{it} \text{ для } i,\ j \in I,\ i \ne j,\ 1 \le \tau \le T; \tag{13}$$

$$x_{it}, y_i, z_{ij}, a_{ij} \in \{0,1\}, i,\ j \in I,\ i \ne j,\ 1 \le t \le T; \tag{14}$$

$$x_{it} = 0 \text{ для } i \in I,\ t > l_i \quad \text{или} \quad t < e_i. \tag{15}$$

Переменная $a_{ij}$ равна 1, если поезд $i$ назначен на более ранний временной интервал, чем поезд $j$. Ограничения (11) и целевая функция (8) гарантируют правильные значения переменных $y_i$ в оптимальном решении. Ограничения (12) и целевая функция (8) гарантируют правильные значения переменных $z_{ij}$ в оптимальном решении. Ограничения (13) гарантируют, что $a_{it}$ равна 1, если поезд $i$ назначен на более ранний временной интервал, чем поезд $j$. Оптимальное решение релаксированной задачи линейного программирования, основанной на задаче целочисленного линейного программирования (8)–(15), дает нам первую нижнюю границу $LB_{LP}$.

**2. Лагранжева нижняя граница**

Определим ориентированный граф предшествования $D=D(I, A)$, где множество вершин $I$ соответствует множеству поездов. Дуга $(i, j)$ принадлежит множеству дуг $A$ тогда и только тогда, когда $A_{ji} > 0$. Предполагается, что граф предшествования не содержит циклов и для каждой вершины существует не более одной выходящей дуги. Другими словами, степень исхода каждой вершины не более единицы. В случае если граф предшествования не обладает этим свойством, можно удалить некоторые дуги с целью получения графа, обладающего требуемым свойством. Однако в этом случае качество нижней границы значительно ухудшается. В дальнейшем предполагается, что граф предшествования обладает указанным свойством.

Релаксируем ограничение (3) и запишем соответствующую функцию Лагранжа:

$$f(x,\lambda) = f(x) + \sum_{t=1}^{T} \lambda_t \Big(\sum_{i\in I} x_{it} - G\Big) = f(x) + \sum_{i\in I}\sum_{t=1}^{T} \lambda_t x_{it} - G\sum_{t=1}^{T} \lambda_t,$$

где $x \in X$ ($X$ – множество допустимых решений, определенное ограничениями (2), (4)–(7)) и заданная $\lambda = (\lambda_1, \lambda_2, \ldots, \lambda_T) \ge 0$.

Пусть $x^0$ является оптимальным решением исходной задачи. Тогда

$$LB_\lambda = \min_{x\in X} f(x,\lambda) \le f(x^0,\lambda) \le f(x^0)$$

и поэтому $LB_\lambda$ – это нижняя граница для исходной задачи для любого $\lambda > 0$.

Запишем функцию Лагранжа в следующем виде:

$$f(x,\lambda) = \bar{c}(\lambda)x - G\sum_{t=1}^{T} \lambda_t = \sum_{t=1}^{T}\sum_{i=1}^{n} \bar{c}^{x}_{it}(\lambda) x_{it} + \sum_{i=1}^{n} \bar{c}^{y}_{i}(\lambda) y_i + \sum_{i,j,i\ne j} \bar{c}^{z}_{ij}(\lambda) z_{ij} - G\sum_{t=1}^{T} \lambda_t,$$

где $\bar{c}^{x}_{it}(\lambda) = \lambda_t,\ 1 \le i \le n,\ 1 \le t \le T$, $\bar{c}^{y}_{i}(\lambda) = \alpha_1,\ 1 \le i \le n$, $\bar{c}^{z}_{ij}(\lambda) = 0$, если $j \notin L_i$, и $\bar{c}^{z}_{ij}(\lambda) = \alpha_2 A_{ij}$, если $j \in L_i$.

Возникает вопрос, как найти минимум функции $f(x, \lambda)$ по $x \in X$. Так как были релаксированы ограничения (3) и наложены определенные ограничения на граф предшествования, эта задача становится относительно простой. Существует алгоритм динамического программирования для указанной задачи асимптотической сложности O($nT$). Опишем данный алгоритм. Так как в графе предшествования нет циклов, то можно перенумеровать вершины этого графа в топологическом порядке, т. е. если существует дуга $(i, j)$, то $i<j$. Соответственно вершинам перенумеруем поезда. Рассмотрим вершину $i$ в графе предшествования. Будем называть деревом, индуцированным этой вершиной, поддерево графа, содержащее все вершины, из которых есть путь в вершину $i$, и все дуги этих путей. Пусть $S_i$ – это множество вершин поддерева, индуцированного вершиной $i$. Тогда рассмотрим следующую функцию, которую будем называть редукцией критерия на множество $S_i$:

$$f(x, \lambda, S_i) = \sum_{t=1}^{T} \sum_{i \in S_i} \bar{c}_{it}^{x}(\lambda) x_{i,t} + \sum_{i \in S_i} \bar{c}_{i}^{y}(\lambda) y_i + \sum_{k \in S_i, j \in S_i, k \neq j} \bar{c}_{kj}^{z}(\lambda) z_{kj} .$$

Пусть $S$ – множество вершин без выходящих дуг. Тогда, учитывая свойства графа предшествования, получаем

$$f(x, \lambda) = \sum_{i \in S} f(x, \lambda, S_i) - G \sum_{t=1}^{T} \lambda_t .$$

Пусть $M(i, t)$ и $Q(i, t)$, $1 \le t \le T$, $i \in I$, – переменные состояния алгоритма динамического программирования. Для каждого расписания редуцируем его целевую функцию на множество $S_i$. Переменная $M(i, t)$ – это минимальное значение такой редуцированной целевой функции на тех расписаниях поездов, в которых поезд $i$ назначен на временной интервал $t$. Если не существует допустимых расписаний с поездом $i$, назначенным на временной интервал $t$, то $M(i, t)=\infty$. Переменная $Q(i, t)$, $1 \le t \le T$, $i \in I$, – это минимальное значение редуцированной целевой функции на тех расписаниях поездов, в которых поезд $i$ назначен на временной интервал не позднее $t$. Если не существует допустимых расписаний с поездом $i$, назначенным не позднее временного слота $t$, то $Q(i,t)=\infty$. Также полагаем, что $Q(i, 0)= \infty$ для всех $i \in I$. Тогда верны следующие рекуррентные соотношения:

$$M(i,t) = \infty,\ i \in I,\ t < e_i \text{ или } t > l_i;$$

$$Q(i,t) = \infty,\ i \in I,\ t < e_i;$$

$$M(i,t) = \bar{c}_{it}^{x} + \min\{\bar{c}_{i}^{y} + \sum_{j \in L_i} \min\{M(j, t),\ Q(j, t-1) + \bar{c}_{ij}^{z} + \bar{c}_{ji}^{z},\ Q(j, T) + \bar{c}_{ij}^{z} + \bar{c}_{ji}^{z}\},$$

$$\sum_{j \in L_i} \min\{M(j,t),\ Q(j, t-1) + \bar{c}_{ij}^{z} + \bar{c}_{ji}^{z}\}\},\ i \in I,\ e_i \le t \le l_i;$$

$$Q(i,t) = \min\{Q(i, t-1),\ M(i,t)\},\ i \in I,\ \ e_i \le t,$$

где $L_i$ – множество предшественников вершины $i$.

Рекуррентное соотношение для переменной $M(i, t)$ учитывает, посещает ли поезд $i$ повторно узел. Если да, то поезда, несущие контейнеры для поезда $i$, могут быть назначены на любой временной интервал, в противном случае поезда, несущие контейнеры для поезда $i$, должны быть назначены не позднее временного интервала поезда $i$. Если поезда назначены на разные временные интервалы, то добавляется стоимость $\bar{c}_{ij}^{z} + \bar{c}_{ji}^{z}$. Оптимальное значение целевой функции вычисляется как

$$f(x_{\min,} \lambda) = \sum_{i \in S} Q(i, T) - G \sum_{t=1}^{T} \lambda_t .$$

Соответствующее расписание может быть найдено с помощью поиска с возвратами по переменным состояния. Асимптотическая сложность алгоритма динамического программирования, основанного на данных рекуррентных соотношениях, равна O($nT$).

Сейчас, когда построен алгоритм для вычисления $LB_\lambda$, возникает вопрос, как выбирать вектор λ лагранжевых множителей.

Начальный вектор λ можно выбрать следующим образом. Пусть $r_{i,t}$=1, если $e_i \le t \le l_i$, и $r_{it}$=0, если $t < e_i$ или $t > l_i$. Тогда полагаем $\lambda_t = \sum_{i \in I} r_{it}$.

Предположим, что имеется начальный вектор λ лагранжевых множителей. Можно искать лучшие значения λ среди векторов λ*, которые вычисляются следующим образом. Зафиксируем $t$. Положим λ*= λ + Δλ, где $\Delta\lambda_h$=0, $h$ не равно $t$ и $\Delta\lambda_t$ принимает поочередно значения 0,2, –0,2, 0,5, –0,5, 1,0, –1,0, 2,0, –2,0, 5,0, –5,0, 10,0, –10,0, 20,0, –20,0, 50,0, –50,0. Для всех таких неотрицательных λ* вычисляется $LB_{\lambda*}$. После этого выбирается вектор λ* с наибольшим значением $LB_{\lambda*}$ или начальное значение λ, если $LB_\lambda$ больше. Эта итерация повторяется для каждой координаты $t$ от 1 до $T$.

### 3. Вычислительный эксперимент

Проведен вычислительный эксперимент на компьютере с процессором Intel Pentium 2.99 GHz CPU и 1 GB оперативной памяти. Программирование осуществлялось на C++ с использованием Visual Studio 2005.

Сначала проводилось сравнение простой нижней границы $LB_{simple}$, полученной в работе Бойсена и др. [1], и лагранжевой нижней границы $LB_{\lambda*}$ (табл. 1).

Таблица 1

Результаты сравнения лагранжевой и простой нижней границы

| Значения $T$, $G$ | Тип интервалов доступности | Тип графа | $LB_{\lambda*} > LB_{simple}$ | $\Delta_{av}$ | $LB_{simple}$ ноль | $LB_{\lambda*}$ ноль |
|---|---|---|---|---|---|---|
| T=15, G=10 | Плотный | Ограниченный | 86 | 0,985 46 | 47 | 13 |
| T=15, G=10 | Плотный | $P$=1/$n$ | 77 | 1,0 | 100 | 23 |
| T=15, G=10 | Плотный | $P$=0,5 | 0 | - | 0 | 16 |
| T=15, G=10 | Свободный | Ограниченный | 0 | - | 45 | 100 |
| T=15, G=10 | Свободный | $P$=1/$n$ | 0 | - | 100 | 100 |
| T=15, G=10 | Свободный | $P$=0,5 | 0 | - | 0 | 100 |
| T=8, G=6 | Плотный | Ограниченный | 99 | 0,949 511 | 0 | 0 |
| T=8, G=6 | Плотный | $P$=1/$n$ | 98 | 0,982 186 | 65 | 2 |
| T=8, G=6 | Плотный | $P$=0,5 | 0 | - | 0 | 0 |
| T=8, G=6 | Свободный | Ограниченный | 84 | 0,816 772 | 5 | 9 |
| T=8, G=6 | Свободный | $P$=1/$n$ | 53 | 0,963 335 | 64 | 46 |
| T=8, G=6 | Свободный | $P$=0,5 | 0 | - | 0 | 9 |

Интервалы доступности поездов генерировались двух типов. Первый тип («плотный») генерировался следующим образом: для трети поездов интервалы доступности генерировались равномерно из отрезка [1, $T$], для трети поездов $e_i$=1 и число $l_i$ генерировалось равномерно из интервала [1, $T$], для трети поездов $l_i$=$T$ и число $e_i$ генерировалось равномерно из интервала [1, $T$]. Второй тип («свободный») генерировался следующим образом: для четных поездов $i$ $e_i$=1 и $l_i$=$T$, для нечетных $i$ интервал доступности был сгенерирован равномерно из интервала [1, $T$].

Далее рассматриваются три типа графов предшествования. Первый тип графа предшествования («ограниченный») – это граф без циклов со степенью исхода вершины не более единицы, второй тип графа – граф, в котором каждая дуга присутствует с вероятностью 1/$n$, где $n$ – число поездов, и третий тип графа предшествования – граф, в котором каждая дуга присутствует с вероятностью 0,5.

Количество контейнеров для дуги было сгенерировано как целое число из интервала [1, 20]. Как и в предыдущих работах, веса в критерии были установлены $\alpha_1$=24 и $\alpha_2$=1.

Для каждого примера вычислялось значение $\Delta = (LB_{\lambda*} - LB_{simple})/LB_{\lambda*}$, где $LB_{\lambda*}$ – лагранжева нижняя граница, а $LB_{simple}$ – простая нижняя граница. Для каждого набора параметров было сгенерировано по 100 примеров. Подсчитано количество примеров, в которых лагранжева нижняя граница больше простой нижней границы. Далее, для тех примеров, для которых лагранжева нижняя граница была лучше простой, подсчитано среднее значение Δ. Также подсчитано количество примеров, в которых лагранжева и простая нижняя границы были равны нулю.

Из табл. 1 видно, что качество лагранжевой нижней границы зависит от типа интервалов доступности следующим образом. Если есть много поездов с длинными интервалами доступности, то качество лагранжевой нижней границы хуже, чем в случае если есть мало таких поездов. Вообще говоря, это поведение лагранжевой границы предсказуемо, так как в случае если интервалы доступности всех поездов равны [1, *T*], лагранжева нижняя граница не может быть больше нуля. Простая нижняя граница не использует информацию об интервалах доступности.

Эксперименты показывают, что лагранжева нижняя граница лучше для ограниченных и разреженных графов. Простая нижняя граница не очень сильна в этих случаях и для большинства примеров равна нулю.

Время вычисления, однако, больше для лагранжевой нижней границы. Для случая *T*=15 и *G*=10 вычисление простой нижней границы требует приблизительно 3 мс, в то время как вычисление лагранжевой нижней границы требует приблизительно 481 мс.

Далее было проведено сравнение лагранжевой нижней границы и нижней границы, основанной на релаксированной задаче линейного программирования. Для каждого примера вычислялось значение $\Delta = (LB_{LP} - LB_{\lambda*})/LB_{LP}$, где $LB_{\lambda*}$ – лагранжева нижняя граница, а $LB_{LP}$ – нижняя граница, основанная на релаксированной задаче линейного программирования. Для каждого набора параметров сгенерировано по 100 примеров. Подсчитано количество примеров, в которых нижняя граница, основанная на релаксированной задаче линейного программирования, была больше лагранжевой нижней границы, и для этих примеров подсчитано среднее значение Δ. Результаты сравнения представлены в табл. 2.

Таблица 2

Сравнение лагранжевой нижней границы и нижней границы, основанной на релаксированной задаче линейного программирования

| Значения *T*, *G* | Тип интервалов доступности | Тип графа | $LB_{\lambda*} < LB_{LP}$ | $\Delta_{av}$ |
|---|---|---|---|---|
| T=8, G=6 | Плотный | Ограниченный | 100 | 0,092 76 |
| T=8, G=6 | Плотный | *P*=1/*n* | 100 | 0,556 141 |
| T=8, G=6 | Плотный | *P*=0,5 | 100 | 0,973 484 |
| T=8, G=6 | Свободный | Ограниченный | 100 | 0,266 339 |
| T=8, G=6 | Свободный | *P*=1/*n* | 100 | 0,805 471 |
| T=8, G=6 | Свободный | *P*=0,5 | 100 | 0,865 58 |

Отметим, что нижняя граница, основанная на релаксированной задаче линейного программирования, заметно лучше на всех рассматриваемых примерах, однако требует значительно большего времени для своего вычисления. В то время как лагранжева нижняя граница требует в среднем 22 мс для своего вычисления, нижняя граница, основанная на релаксированной задаче линейного программирования (вычисляется с помощью CPLEX), требует в среднем 4,158 с для вычисления.

**Заключение**

Рассмотренная в настоящей статье задача оптимизации железнодорожного узла встречается на практике и довольно важна, так как многие страны осуществляют строительство железнодорожных узлов нового типа, что обусловливает использование соответствующих оптимизационных моделей.

Простая нижняя граница была известна для данной задачи из более ранних работ. В статье разработаны две новые нижние границы. Первая из них основана на релаксированной задаче линейного программирования, вторая использует технику лагранжевой релаксации. Вычислительные эксперименты показали, что самой точной нижней границей является нижняя граница, основанная на релаксированной задаче линейного программирования, однако она требует значительно большего времени для своего вычисления. Лагранжева нижняя граница лучше, чем простая нижняя граница, в случае графов специального типа (без циклов и со степенью исхода каждой вершины не более единицы) и разреженных графов, а также когда рассматриваются «плотные» интервалы доступности поездов. Однако лагранжева нижняя граница требует больше времени для своего вычисления по сравнению с простой нижней границей. Асимптотическая сложность алгоритма для простой нижней границы равна $O(n\log(n)+ e + T)$, где $n$ – количество поездов, $e$ – количество дуг в графе предшествования и $T$ – количество временных интервалов. Асимптотическая сложность алгоритма для лагранжевой нижней границы равна $O(nT)$.

Данное исследование было частично поддержано проектами Ф10ФП-001 и Ф10М-071 Белорусского республиканского фонда фундаментальных исследований.

Авторы статьи выражают благодарность Н.Н. Писаруку за важные и полезные замечания, которые значительно улучшили настоящую работу.

**Список литературы**

1. Boysen, N. New bounds and algorithms for the Transshipment Yard Scheduling Problem / N. Boysen, F. Jaehn, E. Pesch // Journal of Scheduling. – 2012. – Vol. 15, I. 4. – P. 499–511.

2. Boysen, N. Scheduling freight trains in rail-rail transshipment yards / N. Boysen, F. Jaehn, E. Pesch // Transportation Science. – 2011. – Vol. 45, I. 2. – P. 199–211.

3. Cordeau, J.F. A survey of optimization models for train routing and scheduling / J.F. Cordeau, P. Toth, D. Vigo // Transportation Science. – 1998. – Vol. 32. – P. 380–404.

4. Macharis, C. Opportunities for OR in intermodal freight transport research: A review / C. Macharis, Y.M. Bontekoning // European Journal of Operational Research. – 2004. – Vol. 153. – P. 400–416.

5. Bontekoning,Y.M. Is a new applied transportation research field emerging? A review of intermodal rail-truck freight transport literature / Y.M. Bontekoning, C. Macharis, J.J. Trip // Transportation Research Part A: Policy and Practice. – 2004. – Vol. 38. – P. 1–24.

6. Crainic, T.G. Intermodal transport / T.G. Crainic, K.H. Kim // In Transportation, Handbooks in Operations Research and Management Science 14, eds.: C. Barnhart, G. Laporte. – North-Holland, 2007. – P. 467–538.

7. Hoogeveen, J.A. Stronger Lagrangian bounds by use of slack variables: applications to machine scheduling problems / J.A. Hoogeveen, S.L. van de Velde // Mathematical Programming. – 1995. – Vol. 70. – P. 173–190.

8. Fisher, M.L. A dual algorithm for the one-machine scheduling problem / M.L. Fisher // Mathematical Programming. – 1976. – Vol. 11. – P. 229–251.

9. Van de Velde, S.L. Dual decomposition of a single-machine scheduling problem / S.L. van de Velde // Mathematical Programming . – 1995. – Vol. 69. – P. 413–428.

10. Fischetti, M. An Additive Bounding Procedure for Combinatorial Optimization Problems / M. Fischetti, P. Toth // Operations Research. – 1989. – Vol. 37, № 2. – P. 319–328.

**Поступила 01.06.12**

[1]*Объединенный институт проблем информатики НАН Беларуси, Минск, Сурганова, 6*
*e-mail: barketau@mail.ru*

[2]*Университет Бремена, Германия*

[3]*Университет Зигена, Германия*

**M.S. Barketau, H. Kopfer, E. Pesch**

## LOWER BOUNDS OF THE CRITERIUM IN THE PROBLEM OF ASSIGNEMENT OF TRAINS ON THE TIME SLOTS

In this paper we consider the container transshipment problem at a railway hub. New lower bounds are developed. One lower bound is based on the linear relaxation of the integer linear formulation, and the other lower bound is based on the Lagrangian relaxation technique.