

ЛАГРАНЖЕВЫ РЕЛАКСАЦИИ В ЗАДАЧЕ
ВЫБОРА ОПТИМАЛЬНОГО СОСТАВА СИСТЕМЫ
ТЕХНИЧЕСКИХ СРЕДСТВ

Ю. А. Кочетов, М. Г. Пашенко

Задачи выбора оптимального состава системы технических средств давно привлекают к себе внимание исследователей [1]. Интерес к ним объясняется не только широкими приложениями, но и тем, что задачи этого класса, как правило, оказываются *NP*-трудными и для их решения приходится разрабатывать специализированные оптимизационные алгоритмы. В простейшей постановке эти задачи являются обобщением классической задачи о размещении производства, которой посвящена обширная литература [2].

В работе предлагаются алгоритмы решения задачи, основанные на Лагранжевых релаксациях. Рассмотрены две релаксации и соответствующие им двойственные задачи. Показано, что одна из двойственных задач полиномиально разрешима, а вторая при фиксированных множителях Лагранжа *NP*-трудна. Обе релаксации используются для построения верхних и нижних оценок, причем полиномиальные оценки оказываются существенно менее точными. Предлагается комбинированный алгоритм решения задачи, использующий последовательно обе релаксации и имеющий относительную погрешность в среднем не более одного процента. Приводятся результаты численных экспериментов.

1. Постановка задачи

Пусть множество $I = \{1, 2, \dots, m\}$ задает исходный ряд образцов технических средств. Исходный ряд включает в себя номера образцов, которые выпускаются серийно, либо могут быть выпущены в ближайшем будущем. Системой технических средств будем называть произвольный набор образцов из исходного ряда.

Технические средства предназначены для выполнения определенных работ. Совокупность этих работ будем называть областью применения системы и обозначать множеством $J = \{1, 2, \dots, n\}$. Считаем, что для каждой работы существуют технические средства, предназначенные для ее выполнения и каждое техническое средство может использоваться только для выполнения работ области применения. Процесс выполнения работ разбит на этапы. На

первом этапе выполняются работы из множества $J_1 \subset J$, на втором - $J_2 \subset J$ и т.д. Будем считать, что

$$J = \bigcup (J_l, l=1, \dots, L), \quad J_l \cap J_k = \emptyset, \quad k \neq l, \quad |J_l| = n_l.$$

Задача состоит в том, чтобы найти состав системы, который выполняет все работы с минимальными суммарными затратами.

Математическая постановка может быть записана в виде задачи линейного частично-целочисленного программирования. Для каждого образца $l \in I$ обозначим через:

C_i^0 - начальные затраты на научные исследования, опытно-конструкторские работы и подготовку производства;

C_i - затраты на производство одного изделия;

V_i - максимально возможный объем производства изделий;

S_{il} - доля потерь на l -м этапе;

V_i^0 - объем изделий, уже имеющихся в наличии (начальный состав системы);

P_{ij} - количество средств, требующееся на выполнение работы $J \in J$;

C_{ij} - затраты на их эксплуатацию.

Переменные величины $(X_i), (V_i), (X_{ij})$ имеют следующий смысл:

$X_i = \begin{cases} 1, & \text{если } i\text{-й образец включен в состав системы,} \\ 0, & \text{в противном случае;} \end{cases}$

$V_i \geq 0$ - объем производства i -го изделия;

$X_{ij} \geq 0$ - доля работы J -го вида, выполняемая изделиями i -го образца.

Сформулированная задача (обозначим ее P) записывается следующим образом:

$$Z_P = \min \sum_{i \in I} (C_i^0 X_i + C_i V_i + \sum_{j \in J} C_{ij} X_{ij}), \quad (1)$$

$$\sum_{i \in I} X_{ij} = 1, \quad j \in J, \quad (2)$$

$$(P) \quad \sum_{i \in I} X_i \leq K, \quad (3)$$

$$\sum_{j \in J} P_{ij} X_{ij} \leq V_i^0 + V_i - \sum_{t=1}^{l-1} S_{it} \cdot \sum_{j \in J_t} P_{ij} X_{ij}, \quad i \in I, \quad l=1, \dots, L, \quad (4)$$

$$0 \leq V_i \leq V_i, \quad i \in I, \quad (5)$$

$$0 \leq X_{ij} \leq X_i, \quad i \in I, \quad j \in J, \quad (6)$$

$$X_i \in \{0, 1\}, \quad i \in I. \quad (7)$$

Целевая функция выражает суммарные затраты на создание и функционирование системы. Равенства (2) гарантируют выполнение всех работ области применения. Ограничение (3) устанавливает потолок на число типов используемых технических средств (не более K типов, K - целое чис-

ло). В левой части неравенства (4) стоит выражение для объема средств l -го образца, требующегося на l -м этапе, а в правой части — выражение для имеющегося в наличии объема с учетом потерь средств на предшествующих этапах. Ограничение (5) устанавливает верхнюю границу на возможные объемы производства изделий, а неравенство (6) запрещает использование технических средств, не включенных в состав системы.

2. Нижние оценки

Для получения нижних оценок в задачах частично-целочисленного программирования традиционно используется линейная релаксация, при которой условие булевости переменных заменяется на требование их принадлежности единичному отрезку:

$$0 \leq x_i \leq 1, \quad i \in I. \quad (7')$$

Обозначим такую задачу через P' , а значение ее целевой функции через Z'_P . В дальнейшем штрихом (') всегда будем обозначать переход к линейной релаксации. Очевидно, что $Z'_P \leq Z_P$. Задача P' является задачей линейного программирования, и значение Z'_P может быть найдено за полиномиальное время. К сожалению, Z'_P , как правило, не совпадает с Z_P , и величина разности $Z_P - Z'_P$, называемая разрывом двойственности, может достигать значительных размеров [2].

Используя различные эквивалентные переформулировки задачи, можно улучшить оценку Z'_P . Заменяем в исходной задаче ограничения (4), (5) на условия

$$\sum_{j \in J_l} p_{ij} x_{ij} \leq v_i^0 x_i + v_i - \sum_{t=1}^{l-1} S_{it} \cdot \sum_{j \in J_t} p_{ij} x_{ij}, \quad i \in I, \quad l=1, \dots, L, \quad (4')$$

$$0 \leq v_i \leq V_i x_i, \quad i \in I, \quad (5')$$

и обозначим полученную задачу через P_x . Очевидно, что задачи P и P_x эквивалентны, но $Z'_P \leq Z'_x$, так как область допустимых значений в задаче P'_x меньше, чем в задаче P' .

З а м е ч а н и е. Нетрудно убедиться [2], что для любого $N > 0$ существует семейство исходных данных, на котором одновременно $Z'_P > Z'_x \cdot N$ и $Z_P > Z'_x \cdot N$.

Более широкие возможности для получения нижних оценок открываются при использовании Лагранжевых релаксаций [3]. Рассмотрим две релаксации задачи, при которых в целевую функцию заносятся ограничения (2) и (4'), (5') соответственно. В первом случае имеем задачу $LR1(\lambda)$:

$$Z_{LR1}(\lambda) = \min \sum_{i \in I} (c_i^0 x_i + c_i v_i - \sum_{j \in J} (\lambda_j - c_{ij}) x_{ij}) + \sum_{j \in J} \lambda_j, \quad (LR1(\lambda))$$

при ограничениях (3), (4'), (5'), (6), (7).

Во втором случае - задачу $LR2(\varphi, \psi)$:

$$Z_{LR2}(\varphi, \psi) = \min \sum_{i \in I} \left\{ (c_i^0 - \sum_{l=1}^L v_l^0 \varphi_{il} - \psi_l v_l) x_i + (c_i - \sum_{l=1}^L \varphi_{il} + \psi_l) v_i + \sum_{j \in J} (c_{ij} + p_{ij}(\varphi_{il} + \sum_{t=l+1}^L s_{it} \varphi_{it})) x_{ij} \right\}$$

($LR2(\varphi, \psi)$)

при ограничениях (2), (3), (6), (7).

Известно [3], что при любом выборе множителей Лагранжа $\lambda \geq 0$, $\varphi \geq 0$, $\psi \geq 0$, значения $Z_{LR1}(\lambda)$ и $Z_{LR2}(\varphi, \psi)$ являются нижними оценками для величины Z_P . Решения двойственных задач

$$(D) \quad Z_D = \max_{\lambda \geq 0} Z_{LR1}(\lambda),$$

$$(C) \quad Z_C = \max_{\varphi \geq 0, \psi \geq 0} Z_{LR2}(\varphi, \psi)$$

позволяют получать наилучшие из таких оценок.

Т е о р е м а 1. Справедливы соотношения

$$Z'_P \leq Z'_P = Z_D \leq Z_C \leq Z_P.$$

Д о к а з а т е л ь с т в о. Первое и последнее неравенства очевидны. Убедимся в том, что $Z'_P = Z_D$. Из соотношений двойственности для линейного программирования $Z'_P = Z'_D$ и нужно проверить, что $Z'_D = Z_D$. Покажем, что $Z_{LR1}(\lambda)' \geq Z_{LR1}(\lambda)$, откуда и будет следовать требуемое равенство. Обозначим через $a_i(\lambda)$ оптимальное значение целевой функции в задаче:

$$a_i(\lambda) = \max \left\{ \sum_{j \in J} (\lambda_j - c_{ij}) x_{ij} - c_i v_i - c_i^0 \right\}$$

при ограничениях (4), (5) и $0 \leq x_{ij} \leq 1$, $j \in J$, $i \in I$, а через $Z_{RN}(a)$ - оптимальное значение целевой функции в простейшей задаче о ранце:

$$(RN(a)) \quad Z_{RN}(a) = \max \sum_{i \in I} a_i(\lambda) x_i, \quad \sum_{i \in I} x_i \leq K, \quad x_i \in (0, 1), \quad i \in I.$$

Тогда $Z_{LR1}(\lambda) = \sum_{j \in J} \lambda_j - Z_{RN}(a)$.

Пусть $(x_i^*), (v_i^*), (x_{ij}^*)$ - оптимальное решение задачи $LR1(\lambda)'$. Построим по данному решению допустимое решение $(\hat{x}_i), (\hat{v}_i), (\hat{x}_{ij})$ задачи $LR1(\lambda)$. Определим величины $\hat{a}_i(\lambda)$ равенствами:

$$\hat{a}_i(\lambda) = \sum_{j \in J} (\lambda_j - c_{ij}) x_{ij}^* - c_i v_i^* - c_i^0, \quad i \in I,$$

$$v_i = \begin{cases} 0, & \text{если } x_i^* = 0, \\ v_i^* / x_i^*, & \text{если } x_i^* > 0, \end{cases} \quad x_{ij} = \begin{cases} 0, & \text{если } x_{ij}^* = 0, \\ x_{ij}^* / x_i^*, & \text{если } x_i^* > 0. \end{cases}$$

Возьмем в качестве \hat{x}_i оптимальное решение задачи $RN(\hat{a})$ и положим $\hat{v}_i = \hat{x}_i v_i$, $\hat{x}_{ij} = \hat{x}_i x_{ij}^*$, $i \in I$, $j \in J$. Легко проверить, что построенное решение допустимо в $LR1(\lambda)$ и кроме того

$$\sum_{j \in J} \lambda_j - Z_{LR1}(\lambda)' = \sum_{i \in I} \left\{ \sum_{j \in J} (\lambda_j - c_{ij}) x_{ij}^* - c_i v_i^* - c_i^0 x_i^* \right\} = \sum_{i \in I} \hat{a}_i(\lambda) x_i^*.$$

Так как $\sum_{i \in I} x_i^* \leq K$ и $0 \leq x_i^* \leq 1$, то x_i^* — допустимое решение задачи $RN(\hat{a})'$, но $Z_{RN}(\hat{a})' = Z_{RN}(\hat{a})$ при целых K , откуда $\sum_{i \in I} \hat{a}_i(\lambda) \hat{x}_i \geq \sum_{i \in I} \hat{a}_i(\lambda) x_i^*$ и $Z_{LR1}(\lambda)' \geq Z_{LR1}(\lambda)$.

Покажем, что $Z_D \leq Z_C$. Так как $Z_D = Z'_x$, а для задач линейного программирования P'_x и C' справедливо равенство $Z'_x = Z'_C$, то $Z_D = Z'_x = Z'_C \leq Z_C$. ■

С л е д с т в и е. Задача D разрешима за полиномиальное время от длины записи исходных данных.

З а м е ч а н и е. Если при формулировке задачи $LR1(\lambda)$ вместо условий (4'), (5') использовать ограничения (4), (5), то получаемая таким образом задача $LRO(\lambda)$ будет эквивалентна задаче $LR1(\lambda)$. Соответствующие двойственные задачи D_0 и D также будут эквивалентны, и утверждение теоремы можно сформулировать в виде: $Z'_p \leq Z'_x = Z_{D_0} \leq Z_C \leq Z_p$.

Из доказательства теоремы следует, что за полиномиальное время можно не только вычислить величину Z_D , но и найти множители Лагранжа λ^* , для которых $Z_D = Z_{LR1}(\lambda^*)$. Пусть μ — оптимальные двойственные оценки, соответствующие в задаче P'_x равенствам (2). Тогда $Z'_x = Z_{LR1}(\mu)' \leq Z_D$, но $Z'_x = Z_D$, откуда $\lambda^* = \mu$.

Решение двойственных задач C и D , как правило, осуществляется методами субградиентной оптимизации [4], в которых на каждой итерации решается релаксированная задача и по ее оптимальному решению корректируются множители Лагранжа. Число итераций в среднем не превышает $10n$. Решение задачи $LR1(\lambda)$, как следует из доказательства теоремы, сводится по сути к вычислению величин $\hat{a}_i(\lambda)$, $i \in I$, что равносильно решению задачи линейного программирования. Задача $LR2(\phi, \psi)$ после очевидных преобразований сводится к классической NP -трудной задаче размещения производства, для решения которой приходится использовать вычислительные схемы типа "ветвей и границ" [1].

С точки зрения теории NP -полноты задачи C и P одинаково трудны, но практически задачу C решать проще и в первую очередь потому, что при фиксированных булевых переменных (x_i) значения непрерывных переменных в задаче $LR2(\phi, \psi)$ определяются за линейное время, а в задаче P для этого приходится решать задачу линейного программирования, для которой неизвестно строго полиномиального алгоритма. Кроме того, для задачи $LR2(\phi, \psi)$ уже создан [1] алгоритм ветвей и границ, хорошо работающий на средних размерностях ($n, m \leq 100$), в то время как для исходной задачи создание такого алгоритма связано с определенными трудностями.

3. Алгоритм решения задачи $LR1(\lambda)$

Т е о р е м а 2. При фиксированных множителях $\lambda_j, j \in J$, оптимальное решение задачи $LR1(\lambda)$ может быть найдено с трудоемкостью $O(m \cdot \log m + m \cdot n \cdot (L + \log n))$.

Д о к а з а т е л ь с т в о теоремы вытекает из нижеследующих лемм. Пусть, как и прежде,

$$a_l(\lambda) = \max \left\{ \sum_{j \in J} (\lambda_j - c_{lj}) x_{lj} - c_l v_l - c_l^0 \right\}$$

при ограничениях (4), (5) и $0 \leq x_{lj} \leq 1, j \in J, l \in I$.

При заданных значениях $a_l(\lambda), l \in I$, задача $LR1(\lambda)$, как уже отмечалось, сводится к простейшей задаче о ранце $RN(a)$, которая решается с трудоемкостью $O(m \cdot \log m)$, и для доказательства теоремы остается указать алгоритм вычисления величин $a_l(\lambda), l \in I$. Так как значения $a_l(\lambda)$ определяются независимо друг от друга, то индекс l в дальнейшем будем опускать. Определим функции $F_l(v), v \geq 0, l=1, \dots, L$ равенствами:

$$F_l(v) = \max \sum_{t=1}^L \sum_{j \in J_t} c_j x_j,$$

$$\sum_{j \in J_t} p_j x_j \leq v - \sum_{r=1}^{t-1} S_r \cdot \sum_{j \in J_r} p_j x_j, \quad t=1, \dots, L,$$

$$0 \leq x_j \leq 1, \quad j \in J_t, \quad t=1, \dots, L,$$

где $c_j = \lambda_j - c_{lj}$, и представим $a(\lambda)$ в следующем виде:

$$a(\lambda) = \left(\max_v \{ F_1(v) - c(v - v^0)^+ \} - c^0 \right), \quad 0 \leq v \leq V + v^0,$$

где $x^+ = \max(0, x)$.

Функции $F_l(v), l=1, \dots, L$ являются вогнутыми кусочно-линейными функциями. Ниже будет показано, что они имеют не более $2n$ точек излома и, значит, величина $a(\lambda)$ может быть найдена по известной функции $F_1(v)$ за $O(\log n)$ операций.

Л е м м а 1. Для функций $F_l(v), l=1, \dots, L$ справедливы рекуррентные соотношения:

$$F_L(v) = G_L(v),$$

$$F_l(v) = \max_{0 \leq u \leq v} (G_l(u) + F_{l+1}(v - S_l u)), \quad l=1, \dots, L-1,$$

где функции $G_l(v), v \geq 0$, определяются равенствами:

$$G_l(v) = \max \sum_{j \in J_l} c_j x_j, \quad \sum_{j \in J_l} p_j x_j \leq v, \quad 0 \leq x_j \leq 1, \quad j \in J_l.$$

Д о к а з а т е л ь с т в о легко проводится обычными для динамического программирования рассуждениями.

Приведенные рекуррентные соотношения позволяют для каждого $v \geq 0$ определить значение функции $F_I(v)$ по известным функциям $F_{I+1}(v)$ и $G_I(v)$. Так как параметр v является непрерывным, то вычислить таким способом все значения функции $F_I(v)$ не представляется возможным. Тем не менее, специальный вид функций $F_I(v)$, а именно их кусочно-линейность, позволяет задавать функции их значениями в точках излома и тем самым вычислять $F_I(v)$ только конечное число раз.

Л е м м а 2. Если $F_I(v^0) = G_I(u^0) + F_{I+1}(v^0 - S_I u^0)$ и $T_I = 1 - S_I \neq 1$, то для достаточно малых $\delta > 0$ верно равенство

$$F_I(v^0 + \delta) = F_I(v^0) + \gamma \delta,$$

$$\gamma = \begin{cases} \alpha, & \text{если } S_I \alpha \geq \beta, \\ \beta + T_I \alpha, & \text{если } S_I \alpha < \beta \text{ и } u^0 = v^0, \\ \beta / S_I, & \text{если } S_I \alpha < \beta \text{ и } u^0 < v^0, \end{cases}$$

где α, β — правые производные функций $F_{I+1}(\mu)$, $G_I(u)$ в точках $\mu = \mu^0 - S_I u^0$, $u = u^0$.

Д о к а з а т е л ь с т в о. Представим функцию $F_I(v)$ в виде

$$F_I(v) = \max_{u, \mu \geq 0} (G_I(u) + F_{I+1}(\mu)), \quad S_I u + \mu \leq v, \quad 0 \leq u \leq v,$$

и выделим два случая: $u^0 = v^0$ и $u^0 < v^0$.

а) Пусть $u^0 < v^0$ и $F_I(v^0 + \delta) = G_I(u^*) + F_{I+1}(\mu^*)$. Рассмотрим область изменения переменных (μ, u) при увеличении v^0 на δ (см. рис. 1). Очевид-

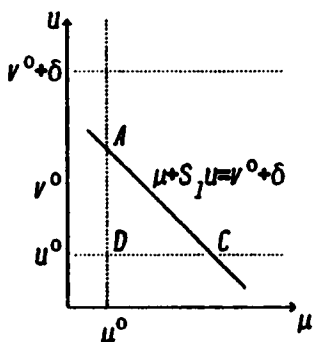


Рис. 1

$$\begin{aligned} A &= (\mu^0, u^0 + \delta / S_I), \\ B &= (\mu^0 + T_I \delta, v^0 + \delta), \\ C &= (\mu^0 + \delta, u^0), \\ D &= (\mu^0, u^0). \end{aligned}$$

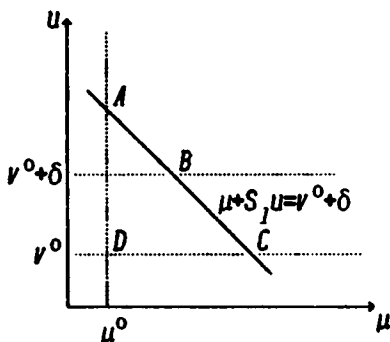


Рис. 2

но, что точка (μ^*, u^*) лежит на прямой $S_I u + \mu = v^0 + \delta$. В силу вогнутости функций $G_I(u)$ и $F_{I+1}(\mu)$ точка (μ^*, u^*) не выходит из отрезка AC . При достаточно малых $\delta > 0$ функции $G_I(u)$ и $F_{I+1}(\mu)$ можно считать линейными на DA и DC . Сумма двух линейных функций тоже линейна. Максимум линейной функции достигается на границе и можно считать, что (μ^*, u^*) совпадает либо с точкой A , либо с точкой C . Приращение $F_I(v)$ при переходе от точки D к точке C равно:

$$\Delta F_C = G_I(u^0) + F_{I+1}(\mu^0 + \delta) - G_I(u^0) - F_{I+1}(\mu^0) = \alpha \delta.$$

Аналогично

$$\Delta F_A = G_I(u^0 + \delta/S_I) + F_{I+1}(\mu^0) - G_I(u^0) - F_{I+1}(\mu^0) = \beta\delta/S_I.$$

Таким образом, если $\alpha \neq \beta/S_I$, то (μ^*, u^*) совпадает с точкой C и $\gamma = \alpha$. В противном случае $\gamma = \beta/S_I$.

б) Пусть $u^0 = v^0$. В этом случае точка A лежит выше прямой $u = v^0 + \delta$, точка (μ^*, u^*) принадлежит отрезку BC (рис. 2) и

$$\Delta F_B = G_I(v^0 + \delta) + F_{I+1}(\mu^0 + T_I \delta) - G_I(v^0) - F_{I+1}(\mu^0) = (\beta + T_I \alpha) \delta.$$

С л е д с т в и е. Функции $F_I(v)$ имеют не более $2n$ точек излома.

Д о к а з а т е л ь с т в о. Пусть функции $G_I(u)$ и $F_{I+1}(\mu)$ имеют соответственно n_I и q_{I+1} точек излома. Будем говорить, что точка излома функции $F_I(v)$ является:

- точкой излома 1-го типа, если для правой производной функции $F_I(v)$ в этой точке верно равенство $\gamma = \alpha$;

- точкой излома 2-го типа, если $\gamma = \beta + T_I \alpha$;

- точкой излома 3-го типа, если $\gamma = \beta/S_I$.

Рассмотрим три случая.

а) Пусть v^0 - точка излома 1-го типа. Тогда $S_I \alpha \geq \beta$,

$$F_I(v^0 + \delta) = G_I(u^0) + F_{I+1}(\mu^0 + \delta)$$

и следующей точке излома v^1 соответствует точка излома функции $F_{I+1}(v)$. Точка v^1 может быть точкой 1-го или 3-го типа, но не может быть точкой 2-го типа, так как $v^1 = v^0 + \delta > u^0 = u^1$.

б) Пусть v^0 - точка излома 2-го типа. Тогда $u^0 = v^0$, $S_I \alpha < \beta$ и

$$F_I(v^0 + \delta) = G_I(u^0 + \delta) + F_{I+1}(\mu^0 + T_I \delta).$$

Точка v^1 определяется по точкам излома либо функции $G_I(u)$, либо функции $F_{I+1}(v)$ и может быть точкой излома 1-го или 2-го типа, причем переходу в точку 1-го типа соответствует точка излома функции $G_I(u)$.

в) Пусть v^0 - точка излома 3-го типа. Тогда $u^0 < v^0$, $S_I \alpha < \beta$ и

$$F_I(v^0 + \delta) = G_I(u^0 + \delta/S_I) + F_{I+1}(\mu^0).$$

Точка v^1 определяется либо точкой излома функции $G_I(u)$, либо равенством

$$v^0 + \delta = u^0 + \delta/S_I. \quad (*)$$

В первом случае точка v^1 может быть точкой излома 1-го или 3-го типа, во втором случае - 2-го типа. Отметим, что если раньше точки излома функции $F_I(v)$ определялись точками излома функций $F_{I+1}(v)$ и $G_I(u)$, то в последнем случае появляется дополнительная точка излома, определяемая равенством (*). Убедимся, что число дополнительных точек излома не превосходит n_I .

Действительно, дополнительная точка излома появляется при переходе от точки 3-го типа к точке 2-го типа. Для того, чтобы снова оказаться в точке 3-го типа надо пройти через точку 1-го типа, а сделать это из

точки 2-го типа можно только пройдя через точку излома функции $G_j(u)$. Таким образом, число точек излома функции $F_l(v)$ не превосходит $n_l + n_l + Q_{l+1}$, откуда

$$Q_1 \leq 2n_1 + Q_2 \leq 2n_1 + 2n_2 + Q_3 \leq \dots \leq 2n. \quad \blacksquare$$

З а м е ч а н и е. Если $S_l = 0$ для некоторого l , то в условиях леммы 2 величина γ всегда равна $\lambda + \beta$ и дополнительные точки излома не появляются.

Конструктивное доказательство последнего утверждения позволяет построить алгоритм вычисления функции $F_l(v)$. Алгоритм состоит из L шагов, на каждом из которых по уже известным функциям $F_{l+1}(v)$ и $G_l(u)$ строится функция $F_l(v)$. Шаг начинается с построения функции $G_l(u)$. Напомним, что $G_l(u)$ является задачей о ранце с непрерывными переменными (u — объем ранца), точки излома функции $G_l(u)$ определяются с трудоемкостью $O(n_l)$, а построение самой функции $G_l(u)$, $u \geq 0$, требует не более $O(n_l \cdot \log n_l)$ операций. По известным $F_{l+1}(v)$ и $G_l(u)$ функция $F_l(v)$ строится за линейное время $O(n_l + Q_{l+1})$, и так как для определения $F_l(v)$ необходимо вычислять все функции $F_j(v)$, $j=1, \dots, L$, то суммарная трудоемкость построения $F_l(v)$ оценивается величиной

$$\sum_{l=1}^L O(n_l \cdot \log n_l + n_l + Q_{l+1}) \leq O(n \cdot \log n + L \cdot n),$$

что и доказывает теорему 2. \blacksquare

4. Построение допустимого решения исходной задачи

Пусть $(x_I^*), (v_I^*), (x_{ij}^*)$ — оптимальное решение задачи $LR1(\lambda)$ и

$$\Delta_f(\lambda) = \sum_{i \in I} x_{ij}^* - 1, \quad j \in J.$$

Если $\Delta_f(\lambda) = 0$, т.е. равенства (2) верны для (x_{ij}^*) , то $(x_i^*), (v_i^*), (x_{ij}^*)$ — оптимальное решение исходной задачи и $Z_{LR1}(\lambda)$ совпадает с Z_P . Выполнение условий (2) означает, что в задаче P_x отсутствует разрыв двойственности. Чаще всего, $Z_{LR1}(\lambda^*) = Z_D < Z_P$, но величина

$$\delta = \sum_{i \in J} |\Delta_f(\lambda^*)|$$

достаточно мала и решение $(x_i^*), (v_i^*), (x_{ij}^*)$ удастся достроить до допустимого решения исходной задачи с помощью процедур координатного спуска. При λ близких к λ^* равенства (2) нарушаются только для некоторых $j \in J$, и трудоемкость такой процедуры практически оказывается очень низкой. Обозначим через H_D алгоритм, использующий эту процедуру в ходе субградиентной оптимизации и выбирающий из полученных решений наилучшее. Близкий по смыслу алгоритм рассматривался в [5].

Обозначим через $P(\Delta)$ задачу, которая отличается от исходной тем,

что равенства (2) заменены на

$$\sum_{i \in I} x_{ij} = 1 + \Delta_j(\lambda), \quad j \in J.$$

Легко проверить, что оптимальное решение задачи $LR1(\lambda)$ является оптимальным решением и в задаче $P(\Delta)$. Алгоритм H_D просматривает оптимальные решения задач $P(\Delta)$ при разных Δ и достраивает их до допустимого решения задачи P . Если задачи P и $P(\Delta)$ "близки" друг к другу, то оптимальное решение одной из них является хорошим приближением для другой.

Алгоритм H_D рассматривает различные наборы (x_i^*) , что, разумеется, не гарантирует получения оптимального набора задачи P . Более осторожной стратегией, чем в алгоритме H_D , является схема покомпонентного составления набора (x_i^*) , на каждом шаге которой одна из переменных x_i полагается равной 1. Ниже приводятся четыре правила выбора такой переменной:

1. Положить $x_i = 1$, если $a_i(\lambda^*) = \max (a_k(\lambda^*), k \in I)$.

2. Определить работу Γ , для которой

$$\sum_{i \in I} x_{ir}^* = \max \left(\sum_{i \in I} x_{ij}^*, j \in J \right)$$

и положить $x_i = 1$, если

$$\sum_{j \in J} x_{ij}^* = \max \left(\sum_{j \in J} x_{kj}^* \mid k \in I, x_{kr}^* > 0 \right).$$

3. Положить $x_i = 1$, если $\sum_{j \in J} x_{ij}^* = \max \left(\sum_{j \in J} x_{kj}^*, k \in I \right)$.

4. Положить $x_i = 1$, если номер i наиболее часто входил в состав оптимального набора (x_i^*) задачи $LR1(\lambda)$ в ходе субградиентной оптимизации.

Правило 1 наиболее широко применяется в методах ветвей и границ для задачи размещения с ограничениями на объемы производства [6]. Аналог правила 2 используется в наиболее быстром точном алгоритме для задач о покрытии и разбиении [7]. Правило 3 хорошо зарекомендовало себя в простейшей задаче размещения производства.

Обозначим через H_q алгоритм, использующий q -е правило выбора и делающий не более K шагов. На каждом шаге выбирается переменная x_i и полагается равной 1. Затем производится корректировка множителей Лагранжа и к каждому решению задачи $LR1(\lambda)$ применяется алгоритм H_D . Результатом работы алгоритма H_q является наилучшее из построенных допустимых решений.

Приведенные алгоритмы базируются на решении задачи $LR1(\lambda)$, хотя нетрудно построить аналогичные конструкции и для релаксации $LR2(\phi, \psi)$. Обозначим через H_C аналог алгоритма H_D , основанный на релаксации $LR2(\phi, \psi)$. Понятно, что алгоритм H_C уже не будет иметь полиномиальной трудоемкости, но так как $Z_C \geq Z_D$, то, проигрывая во времени, он может выигрывать в точности.

5. Численный эксперимент

Предложенные алгоритмы тестировались на классе задач следующей размерности:

$$m = 30, \quad n = 30, \quad n_l = 5, \quad l = 1, \dots, L, \quad L = 6.$$

Исходные данные формировались датчиком случайных чисел с равномерным распределением. Параметры задачи выбирались из следующих интервалов:

$c_i \in [1, 5], \quad c_{ij} \in [0, 105], \quad p_{ij} \in [1, 5], \quad c_i^0 = 100, \quad V_i^0 = 2, \quad S_{ij} = 0.1$.
Матрицы (p_{ij}) и (c_{ij}) на 30% заполнялись числом 10^6 . Расчеты проводились при разных значениях параметров K и V ($V_l = V$, см. ограничение (5)). Результаты расчетов сведены в четыре таблицы. В каждой таблице содержится решение десяти тестовых задач для фиксированной пары параметров (K, V) . Колонка (Z_C) содержит значение Z_C в абсолютных величинах. Колонка (Z_D) позволяет судить о величине разрыва двойственности в решаемых задачах. В ней приведены значения относительного отклонения $\varepsilon = Z_D / Z_C \cdot 100\%$. В остальных колонках содержатся значения относительного отклонения для соответствующих эвристических алгоритмов: $\varepsilon_x = (Z_x / Z_C - 1) \cdot 100\%$.

Решение тестовых задач без ограничений (3), (5) содержит в оптимальном наборе (X_i^*) от 5 до 6 ненулевых элементов, объемы производства меняются от 2 до 11 единиц, и разрыв двойственности в среднем составляет около 3% от Z_C . Навылучшие результаты по точности на указанных классах задач принадлежат алгоритму H_C . При решении задачи C оптимальные наборы (X_i^*) часто повторяются, число различных наборов в среднем не превосходит трех, а в половине случаев такой набор единственный. Интересно отметить, что при решении задачи D наблюдается совершенно другая

Таблица 1. $V = 1000 \quad K = 30$

N	Z_C	Z_D	H_C	H_D	H_1	H_2	H_3	H_4
1	925.5	97.7	0.0	2.2	0.0	0.0	0.0	0.0
2	957.2	96.6	0.0	3.4	0.0	0.0	0.0	0.0
3	989.6	96.9	0.1	0.1	0.1	0.1	0.1	0.1
4	921.3	99.2	0.2	0.2	0.2	0.2	0.2	0.2
5	952.9	95.9	0.1	3.3	4.0	0.4	1.1	0.4
6	969.1	97.2	0.0	2.4	0.7	0.7	1.2	0.7
7	927.6	95.4	0.1	1.1	3.0	1.1	1.1	1.1
8	962.9	96.1	0.1	4.3	0.7	4.9	1.4	1.8
9	974.4	95.3	0.1	2.6	0.4	1.5	1.5	1.5
10	944.3	98.3	0.2	1.7	4.0	5.9	7.3	0.2
среднее	96.9		0.1	3.1	1.3	1.5	1.4	0.6

картина. Из множества I выделяется группа номеров (от 10 до 15 элементов) с примерно равными значениями $a_i(\lambda)$, к которым эвристики H_q принимают правила выбора. Оптимальные наборы (x_i^*) почти не повторяются и наилучшее решение эвристики H_q редко совпадает с набором номеров, выбранным по Q -му правилу. Другими словами, рекордное значение находится алгоритмом на промежуточном шаге с помощью процедуры H_D . Алгоритмы H_q в среднем дают небольшое отклонение от оптимума на задачах, где оба или одно из ограничений (3), (5) являются несущественными (табл. 1-3). Ни одна из эвристик H_q не является доминирующей и все они могут сильно ошибаться, когда ограничения (3), (5) одновременно являются "жесткими" (табл. 4). Такой класс задач является наиболее трудным для данных эвристик и на нем целесообразно использовать комбинацию алгоритмов H_D и H_C . Обозначим через H_{DC} алгоритм, работающий по следующей схеме:

1. Решить задачу D и определить множество

$$I_\varepsilon = \{i \in I \mid a_i(\lambda^*) + \varepsilon \geq \max(a_k(\lambda^*), k \in I)\}.$$

2. Решить задачу C на множестве I_ε , используя алгоритм H_C для построения допустимого решения.

Алгоритм H_{DC} на первом шаге формирует множество номеров, с которыми работают эвристики H_q , а на втором шаге вместо правил выбора использует релаксацию LR2 для построения набора (x_i^*) . Такая комбинация релаксаций позволяет использовать достоинства обоих подходов и сглаживает их недостатки. При $\varepsilon = 0.01 \cdot Z_D$ алгоритм H_{DC} имеет те же результаты по точности, что и алгоритм H_C и тратит в среднем около минуты на решение одной тестовой задачи. Среднее время работы алгоритма H_q составляет 40-50 секунд, алгоритма H_C - 2-3 минуты на IBM PC\MT-386.

Таблица 2.

 $V = 5 \quad K = 30$

N	Z_C	Z_D	H_C	H_D	H_1	H_2	H_3	H_4
1	989.4	96.8	0.1	0.1	0.1	0.1	0.1	0.1
2	947.1	98.9	0.3	8.6	0.3	0.3	0.3	0.3
3	954.8	96.3	0.3	7.0	0.3	0.6	0.3	0.3
4	931.8	97.6	2.1	8.4	1.1	1.1	1.1	1.1
5	973.8	96.7	1.2	5.2	4.2	1.2	1.2	1.2
6	939.4	98.2	1.4	1.4	1.4	1.4	1.4	1.4
7	975.5	96.8	1.5	3.7	2.7	2.7	3.7	2.6
8	962.2	96.1	1.0	4.4	2.1	6.0	4.0	2.1
9	932.5	95.5	2.5	2.6	2.6	4.6	5.6	5.6
10	969.9	96.0	0.6	6.9	0.6	0.6	4.8	4.8
среднее		96.9	1.1	4.8	1.5	1.8	2.2	2.0

Таблица 3.

 $V = 1000 \quad K = 3$

N	Z_C	Z_D	H_C	H_D	H_1	H_2	H_3	H_4
1	1013.0	95.3	0.0	0.0	6.3	0.0	0.0	0.0
2	1074.2	93.2	0.0	0.0	0.0	3.2	0.0	0.0
3	1011.0	97.1	0.0	0.0	0.0	8.9	7.2	0.0
4	988.0	95.6	0.4	0.4	0.4	0.4	4.6	0.4
5	1061.8	91.7	0.1	9.6	2.9	0.1	3.5	3.5
6	1050.9	94.0	0.1	1.8	0.2	1.8	0.2	1.8
7	972.0	95.2	0.7	3.0	3.0	6.4	3.0	0.7
8	1142.9	91.7	0.0	7.6	4.2	2.0	1.3	0.7
9	1014.3	93.5	0.0	4.4	2.3	6.9	2.3	2.3
10	1072.8	93.4	0.0	7.1	3.3	5.4	3.7	3.7
среднее	94.1		0.1	3.4	2.3	3.5	2.6	1.3

Таблица 4.

 $V = 7 \quad K = 3$

N	Z_C	Z_D	H_C	H_D	H_1	H_2	H_3	H_4
1	1075.3	94.3	0.0	0.0	0.0	0.0	6.2	0.0
2	1051.3	94.5	0.1	15.1	5.0	0.1	0.1	0.1
3	1023.9	96.1	0.1	0.1	0.3	0.3	0.1	0.1
4	994.1	96.4	0.8	0.8	0.8	0.8	16.4	0.8
5	1156.2	91.9	1.9	6.7	7.7	5.1	2.0	6.7
6	1014.5	95.3	0.4	5.1	5.1	7.2	3.3	3.3
7	971.5	96.1	0.1	3.9	3.9	3.9	3.9	3.9
8	1075.2	91.6	3.2	14.2	15.7	14.2	6.5	10.1
9	1090.8	92.7	1.2	24.8	5.9	4.2	14.3	19.0
10	1035.7	98.3	0.7	27.5	27.5	12.7	27.5	27.5
среднее	94.7		0.8	9.8	7.2	4.8	8.0	7.2

Поступила ред.-изд. отдел

25 марта 1992 г.

Л и т е р а т у р а

1. Береснев В.Л., Гимади Э.Х., Дементьев В.Т. Экстремальные задачи стандартизации. - Новосибирск: Наука, 1978. - 333 с.
2. Krarup J., Pruzan P. Simple plant location problem, survey and synthesis // Eur. J. Oper. Res. - 1983. - V. 12. - P. 36-81.

3. Shapiro J.F. A survey of Lagrangian techniques for discrete optimization // Annals of Discrete Math. - 1979. - V. 5. - P. 113-138.

4. Held M., Wolfe P., Crowder H.D. Validation of subgradient optimization // Math. Program. - 1974. - V. 6. - P. 62-88.

5. Cornuejols G., Sridharan R., Thizy J.M. A comparison of heuristics and relaxation for the capacitated plant location problem // Eur. J. Oper. Res. - 1991. - V. 50. - P. 280-297.

6. Christofides N., Beasley R. Extensions to a Lagrangean relaxation approach for the capacitated warehouse location problem // Eur. J. Oper. Res. - 1984. - V. 12. - P. 19-28.

7. Fisher M.L., Kedia P. Optimal solution of set covering/partitioning problems using dual heuristics // Management Sci. - 1990. - V. 36. - N.6. - P. 674-688.