

**A Lagrangian Based Branch-and-Bound Algorithm
for Production-Transportation Problems**

Takahito Kuno* and Takahiro Utsunomiya

January 20, 1998

ISE-TR-98-150

Institute of Information Sciences and Electronics

University of Tsukuba

Tsukuba, Ibaraki 305, Japan

Phone: +81-298-53-5540, Fax: +81-298-53-5206, E-mail: takahito@is.tsukuba.ac.jp

* The author was partly supported by Grant-in-Aid for Scientific Research of the Ministry of Education, Science, Sports and Culture, Grant No. (C2)09680413.

A Lagrangian Based Branch-and-Bound Algorithm for Production-Transportation Problems

Takahito Kuno* and Takahiro Utsunomiya
Institute of Information Sciences and Electronics
University of Tsukuba

January 1998

Abstract. We propose a branch-and-bound algorithm of Falk-Soland's type for solving the minimum cost production-transportation problem with concave production costs. To accelerate the convergence of the algorithm, we reinforce the bounding operation using a Lagrangian relaxation, which is a concave minimization but yields a tighter bound than the usual linear programming relaxation in $O(mn \log n)$ additional time. Computational results indicate that the algorithm can solve fairly large scale problems.

Key words: Global optimization, production-transportation problem, minimum concave-cost flow problem, branch-and-bound algorithm, Lagrangian relaxation.

1. Introduction

The production-transportation problem is a model for determining both optimal production at some factories manufacturing a single commodity and optimal transportation of the products to warehouses with known demands. Let $G = (M, N, A)$ be a bipartite graph consisting of a set M of m factories, a set N of n warehouses and a set $A = M \times N$ of mn transportation routes. At each factory $i \in M$ the cost of producing y_i units is $f_i(y_i)$ and the production capacity is u_i units. Due to economy of scale, the production cost $f_i : \mathbb{R} \rightarrow \mathbb{R}$ is assumed to be a nonlinear, concave and nondecreasing function. At each warehouse $j \in N$ there is a demand of b_j units. The cost of shipping a unit by route $(i, j) \in A$ is c_{ij} . The problem is then formulated as follows:

$$\left\{ \begin{array}{ll} \text{minimize} & z = \sum_{(i,j) \in A} c_{ij} x_{ij} + \sum_{i \in M} f_i(y_i) \\ \text{subject to} & \sum_{j \in N} x_{ij} \leq y_i, \quad 0 \leq y_i \leq u_i, \quad i \in M \\ & \sum_{i \in M} x_{ij} = b_j, \quad j \in N \\ & x_{ij} \geq 0, \quad (i, j) \in A, \end{array} \right. \quad (1.1)$$

*The author was partially supported by Grant-in-Aid for Scientific Research of the Ministry of Education, Science, Sports and Culture, Grant No. (C2)09680413.

where x_{ij} s and y_i s are variables to be determined.

The problem (1.1) belongs to the class of capacitated minimum concave-cost flow problems. The main characteristic of (1.1) is that the number m of nonlinear concave variables is rather small in comparison with the number mn of linear variables. Over the past five years, several efficient algorithms using this *low-rank concavity* [12] have been proposed to solve the problem with small m [8, 11, 14, 15, 20, 21, 22]. Each of these algorithms is polynomial or pseudo-polynomial in n . Unfortunately, however, it is exponential in m and will be of no practical use if m exceeds five at most. Therefore, branch-and-bound is still thought of as a most effective approach to the problem (1.1) with larger m , as to general minimum concave-cost flow problems. Branch-and-bound algorithms applicable to (1.1) are classified roughly into two types. The first type remarks on the fact that a basic solution is optimal to (1.1), and implicitly enumerates the spanning trees of G [5, 7]. The second type exploits the separability of the objective function and successively improves its linear underestimator as dividing the feasible set. The basis of this approach is found in Falk and Soland [4]. While they did not assume any network structures, Soland converted it later for a production-transportation problem with concave transportation costs [18]. In their recent book [9], Horst et al. have also applied Falk-Soland's algorithm to (1.1). The readers are also referred to [6, 16] for comprehensive reviews on minimum concave-cost flow problems.

In this paper, we develop a branch-and-bound algorithm of the second type to solve the problem (1.1) with every m . The branching operation in our algorithm is similar to that of Soland, which divides the feasible set of nonlinear variables to generate subproblems; but the bounding operation fully exploits the problem structures and is implemented through two stages: the first stage based on a linear programming relaxation and the second stage based on a Lagrangian relaxation. Although the Lagrangian relaxation of each subproblem is a concave minimization, this two-stage bounding operation provides a lower bound for it much tighter than that of Soland in $O(mn \log n)$ additional computational time. In Section 2, both the relaxations are given in detail. Section 3 is devoted to the algorithm incorporating the two-stage bounding operation. Computational results of the algorithm are reported in Section 4. Some final remarks are discussed in Section 5.

2. Relaxations

We assume throughout the paper that both the production capacity u_i at each factory $i \in M$ and the demand b_j at each warehouse $j \in N$ are positive integers; and that the unit transportation cost c_{ij} by each route $(i, j) \in A$ is a nonnegative real number. We also assume that

$$B \equiv \sum_{j \in N} b_j \leq \sum_{i \in M} u_i. \quad (2.1)$$

Otherwise, the problem (1.1) has no feasible solutions. Since the objective function is continuous and the feasible set is a polytope, (1.1) always has an optimal solution under condition (2.1). In addition to this, the concave objective function achieves its minimum at some vertex of the feasible set. By the total unimodularity, all the vertices are integral vectors as long as b_j s and u_i s are integers (see e.g. [2]). Therefore, once we assume (2.1), the problem (1.1) has an integral optimal solution $(\mathbf{x}^*, \mathbf{y}^*)$ such that

$$\sum_{j \in N} x_{ij}^* = y_i^*, \quad i \in M \quad (2.2)$$

because of the monotonicity of f_i s.

Using (2.2), let us rewrite (1.1) as follows:

$$[\text{PTP}] \left\{ \begin{array}{l} \text{minimize} \quad z = f(\mathbf{x}, \mathbf{y}) \equiv \sum_{(i,j) \in A} c_{ij} x_{ij} + \sum_{i \in M} f_i(y_i) \\ \text{subject to} \quad \sum_{j \in N} x_{ij} = y_i, \quad i \in M \\ \sum_{i \in M} x_{ij} = b_j, \quad j \in N \\ \mathbf{x} \geq \mathbf{0}, \quad \mathbf{y} \in Y, \end{array} \right.$$

where $\mathbf{x} = (x_{ij} \mid (i, j) \in A)$ and $\mathbf{y} = (y_i \mid i \in M)$ are the vectors of linear and nonlinear variables, respectively, and

$$Y = [0, u_1] \times \cdots \times [0, u_m].$$

As mentioned in Section 1, we apply a branching operation on the set Y recursively and divide it into a number of subsets Y^k , $k = 1, \dots, r$. The division $\mathcal{Y} = \{Y^k \mid k = 1, \dots, r\}$ we employ is an *integral rectangular partition* of Y , i.e. for each $k = 1, \dots, r$, we have

$$Y^k = I_1^k \times \cdots \times I_m^k; \quad I_i^k = [l_i^k, u_i^k], \quad i \in M,$$

where l_i^k s and u_i^k s are integers; and

$$\bigcup_{k=1}^r Y^k = Y; \quad \text{int}Y^k \cap \text{int}Y^h = \emptyset \quad \text{if } k \neq h.$$

Associated with each partition set $Y^k \in \mathcal{Y}$ we define a subproblem:

$$[\text{P}^k] \left\{ \begin{array}{l} \text{minimize} \quad z = f(\mathbf{x}, \mathbf{y}) \\ \text{subject to} \quad \sum_{j \in N} x_{ij} = y_i, \quad i \in M \\ \sum_{i \in M} x_{ij} = b_j, \quad j \in N \\ \mathbf{x} \geq \mathbf{0}, \quad \mathbf{y} \in Y^k, \end{array} \right.$$

The following are immediate consequences:

Proposition 2.1. (i) *Problem $[\text{P}^k]$ has an optimal solution if and only if*

$$\sum_{i \in M} l_i^k \leq B \leq \sum_{i \in M} u_i^k. \quad (2.3)$$

If (2.3) holds, $[P^k]$ has at least one integral optimal solution $(\mathbf{x}^k, \mathbf{y}^k)$.

- (ii) Let $z^k = f(\mathbf{x}^k, \mathbf{y}^k)$ if (2.3) holds; $z^k = +\infty$ otherwise. Then the optimal value of $[PTP]$ is given by

$$z^* = \min\{z^k \mid k = 1, \dots, r\}.$$

Given a rectangular partition $\mathcal{Y} = \{Y^k \mid k = 1, \dots, r\}$ of Y , let us take a set $Y^k \in \mathcal{Y}$ satisfying condition (2.3) and consider the associated subproblem $[P^k]$. As a beginning, we will present a linear programming relaxation of $[P^k]$, which minimizes the convex envelop of the objective function f as do both Falk-Soland's and Soland's relaxations [4, 18].

2.1. LINEAR PROGRAMMING RELAXATION

The convex envelop of f_i over the interval $I_i^k = [l_i^k, u_i^k]$ is defined by

$$\bar{f}_i(y_i) = \frac{f_i(u_i^k) - f_i(l_i^k)}{u_i^k - l_i^k} (y_i - l_i^k) + f_i(l_i^k), \quad i \in M,$$

which satisfies

$$\bar{f}_i(y_i) \leq f_i(y_i) \text{ if } y_i \in I_i^k; \quad \bar{f}_i(y_i) \geq f_i(y_i) \text{ if } y_i \notin \text{int}I_i^k. \quad (2.4)$$

Hence, replacing f_i by \bar{f}_i in $[P^k]$ for each $i \in M$, we have a linear program yielding a lower bound for z^k . Actually, in Falk-Soland [4] this linear program is used directly as a relaxation of $[P^k]$. In Soland [18], the set of constraints $\mathbf{y} \in Y^k$ is further relaxed into $\mathbf{y} \in Y$, to make a Hitchcock problem of the resulting problem. We adopt a compromise of these relaxations and replace $\mathbf{y} \in Y^k$ by

$$0 \leq y_i \leq u_i^k, \quad i \in M. \quad (2.5)$$

If we delete y_i by substituting $y_i = \sum_{j \in N} x_{ij}$ into (2.5) for each $i \in M$, our relaxation is also reduced to a Hitchcock problem:

$$[\bar{P}] \left\{ \begin{array}{l} \text{minimize} \quad z = \bar{f}(\mathbf{x}) \equiv \sum_{(i,j) \in A} c_{ij}^k x_{ij} + d^k \\ \text{subject to} \quad \sum_{j \in N} x_{ij} \leq u_i^k, \quad i \in M \\ \sum_{i \in M} x_{ij} = b_j, \quad j \in N \\ \mathbf{x} \geq \mathbf{0}, \end{array} \right.$$

where

$$\left. \begin{aligned} c_{ij}^k &= c_{ij} + \frac{f_i(u_i^k) - f_i(l_i^k)}{u_i^k - l_i^k}, \quad (i, j) \in A \\ d^k &= \sum_{i \in M} \left(f_i(l_i^k) - \frac{f_i(u_i^k) - f_i(l_i^k)}{u_i^k - l_i^k} l_i^k \right). \end{aligned} \right\} (2.6)$$

Theorem 2.2. Problem $[\bar{P}]$ has an integral optimal solution $\bar{\mathbf{x}}$. Let

$$\bar{y}_i = \sum_{j \in N} \bar{x}_{ij}, \quad i \in M; \quad \bar{z} = \bar{f}(\bar{\mathbf{x}}).$$

Then $(\bar{\mathbf{x}}, \bar{\mathbf{y}})$ is feasible to [PTP] and we have

$$\bar{z} \leq z^k; \quad f(\bar{\mathbf{x}}, \bar{\mathbf{y}}) \geq z^*.$$

Proof: Follows from the construction of $[\bar{P}]$. \square

We can thus use \bar{z} as a lower bound for z^k and terminate branching at subproblem $[P^k]$ unless \bar{z} is less than the value of each feasible solution of [PTP] in hand. Although the bound \bar{z} might not be so tight as Falk-Soland's, it is tighter than Soland's; and besides, $[P]$ yielding \bar{z} is a simple Hitchcock problem like Soland's. Nevertheless, the bound \bar{z} is not tight enough yet to improve branch-and-bound algorithms of the second type drastically. To solve [PTP] with large m , we have to devise another relaxation of $[P^k]$ yielding a much tighter bound than $[\bar{P}]$.

2.2. LAGRANGIAN RELAXATION

Let us introduce a Lagrangian multiplier vector $\boldsymbol{\lambda} = (\lambda_j \mid j \in N) \geq \mathbf{0}$ for relaxing the set of constraints $\sum_{i \in M} x_{ij} = b_j, j \in N$. By noting that x_{ij} cannot exceed b_j for each $(i, j) \in A$, we write the resulting problem as follows:

$$[L(\boldsymbol{\lambda})] \left\{ \begin{array}{l} \text{minimize} \quad z = \phi(\mathbf{x}, \mathbf{y}; \boldsymbol{\lambda}) \equiv \sum_{(i,j) \in A} c_{ij}(\boldsymbol{\lambda}) x_{ij} + \sum_{i \in M} f_i(y_i) + \sum_{j \in N} \lambda_j b_j \\ \text{subject to} \quad \sum_{j \in N} x_{ij} = y_i, \quad i \in M \\ \quad \quad \quad 0 \leq x_{ij} \leq b_j, \quad (i, j) \in A, \quad \mathbf{y} \in Y^k, \end{array} \right.$$

where

$$c_{ij}(\boldsymbol{\lambda}) = c_{ij} - \lambda_j, \quad (i, j) \in A. \quad (2.7)$$

This is the second relaxation of $[P^k]$ and plays the central role in our algorithm. As is well-known (see e.g. [17]), we have the following:

Lemma 2.3. Let $(\mathbf{x}(\boldsymbol{\lambda}), \mathbf{y}(\boldsymbol{\lambda}))$ denote an optimal solution of $[L(\boldsymbol{\lambda})]$ and let $z(\boldsymbol{\lambda}) = \phi(\mathbf{x}(\boldsymbol{\lambda}), \mathbf{y}(\boldsymbol{\lambda}); \boldsymbol{\lambda})$. Then

$$z(\boldsymbol{\lambda}) \leq z^k, \quad \forall \boldsymbol{\lambda} \in \mathbb{R}^n.$$

The question here is how we should choose a value of λ_i for each $i \in N$ such that $z(\boldsymbol{\lambda}) > \bar{z}$. To answer this, we need to consider a linear programming relaxation of $[\mathbf{L}(\boldsymbol{\lambda})]$. In the same way as we have constructed $[\bar{\mathbf{P}}]$, we can linearize $[\mathbf{L}(\boldsymbol{\lambda})]$ using \bar{f}_i s into

$$\left\{ \begin{array}{l} \text{minimize} \quad \bar{\phi}(\mathbf{x}, \mathbf{y}; \boldsymbol{\lambda}) \equiv \sum_{(i,j) \in A} (c_{ij}^k - \lambda_j) x_{ij} + d^k + \sum_{j \in N} \lambda_j b_j \\ \text{subject to} \quad \sum_{j \in N} x_{ij} \leq u_i^k, \quad j \in N \\ \quad \quad \quad 0 \leq x_{ij} \leq b_j, \quad (i, j) \in A, \end{array} \right. \quad (2.8)$$

where c_{ij}^k s and d^k are defined in (2.6). The dual problem of (2.8) is then written as follows:

$$\left\{ \begin{array}{l} \text{maximize} \quad \psi(\boldsymbol{\mu}, \boldsymbol{\nu}; \boldsymbol{\lambda}) \equiv - \sum_{i \in M} u_i^k \mu_i - \sum_{(i,j) \in A} b_j \nu_{ij} + d^k + \sum_{j \in N} \lambda_j b_j \\ \text{subject to} \quad -\mu_i - \nu_{ij} \leq c_{ij}^k - \lambda_j, \quad (i, j) \in A \\ \quad \quad \quad \boldsymbol{\mu} \geq \mathbf{0}, \quad \boldsymbol{\nu} \geq \mathbf{0}, \end{array} \right. \quad (2.9)$$

where $\boldsymbol{\mu} = (\mu_i \mid i \in M)$ and $\boldsymbol{\nu} = (\nu_{ij} \mid (i, j) \in A)$ are the vectors of dual variables. Let $\bar{\mathbf{x}}(\boldsymbol{\lambda})$ and $(\bar{\boldsymbol{\mu}}(\boldsymbol{\lambda}), \bar{\boldsymbol{\nu}}(\boldsymbol{\lambda}))$ denote optimal solutions of (2.8) and (2.9), respectively. From the duality theorem in linear programming, we have

$$\begin{aligned} \max_{\boldsymbol{\lambda} \in \mathbf{R}^n} \bar{\phi}(\bar{\mathbf{x}}(\boldsymbol{\lambda}); \boldsymbol{\lambda}) &= \max_{\boldsymbol{\lambda} \in \mathbf{R}^n} \psi(\bar{\boldsymbol{\mu}}(\boldsymbol{\lambda}), \bar{\boldsymbol{\nu}}(\boldsymbol{\lambda}); \boldsymbol{\lambda}) \\ &= \max_{\boldsymbol{\mu}, \boldsymbol{\nu}, \boldsymbol{\lambda}} \left\{ \psi(\boldsymbol{\mu}, \boldsymbol{\nu}, \boldsymbol{\lambda}) \mid \begin{array}{l} -\mu_i - \nu_{ij} \leq c_{ij}^k - \lambda_j, \quad (i, j) \in A \\ \boldsymbol{\mu} \geq \mathbf{0}, \quad \boldsymbol{\nu} \geq \mathbf{0} \end{array} \right\}. \end{aligned} \quad (2.10)$$

Lemma 2.4. The right-hand-side of (2.10) has a maximum point $(\bar{\boldsymbol{\mu}}, \bar{\boldsymbol{\nu}}, \bar{\boldsymbol{\lambda}})$ with $\bar{\boldsymbol{\nu}} = \mathbf{0}$.

Proof: Let $(\boldsymbol{\mu}', \boldsymbol{\nu}', \boldsymbol{\lambda}')$ with $\boldsymbol{\nu}' \neq \mathbf{0}$ be a maximum point. Also let

$$\bar{\lambda}_j = \lambda'_j - \sum_{i \in M} \nu'_{ij}, \quad j \in N; \quad \bar{\mu}_i = \mu'_i, \quad i \in M; \quad \bar{\nu}_{ij} = 0, \quad (i, j) \in A.$$

Then we have

$$\begin{aligned} -\bar{\mu}_i - \bar{\nu}_{ij} &\leq c_{ij}^k - \lambda'_j + \nu'_{ij} \leq c_{ij}^k - \lambda'_j + \sum_{h \in M} \nu'_{hj} = c_{ij}^k - \bar{\lambda}_j, \quad (i, j) \in A \\ \psi(\bar{\boldsymbol{\mu}}, \bar{\boldsymbol{\nu}}, \bar{\boldsymbol{\lambda}}) &= - \sum_{i \in M} u_i^k \bar{\mu}_i + d^k + \sum_{j \in N} \bar{\lambda}_j b_j \\ &= - \sum_{i \in M} u_i^k \mu'_i + d^k + \sum_{j \in N} \left(\lambda'_j - \sum_{i \in M} \nu'_{ij} \right) b_j = \psi(\boldsymbol{\mu}', \boldsymbol{\nu}', \boldsymbol{\lambda}'). \end{aligned}$$

This implies that $(\bar{\boldsymbol{\mu}}, \bar{\boldsymbol{\nu}}, \bar{\boldsymbol{\lambda}})$ is a maximum point as well. \square

Note that, if we set $\boldsymbol{\nu} = \mathbf{0}$ in (2.10), it coincides with the dual problem of $[\bar{\mathbf{P}}]$:

$$\left\{ \begin{array}{l} \text{maximize} \quad g(\boldsymbol{\mu}, \boldsymbol{\lambda}) \equiv \sum_{j \in N} b_j \lambda_j - \sum_{i \in M} u_i^k \mu_i + d^k \\ \text{subject to} \quad \lambda_j - \mu_i \leq c_{ij}^k, \quad (i, j) \in A \\ \quad \quad \quad \boldsymbol{\mu} \geq \mathbf{0}. \end{array} \right. \quad (2.11)$$

Hence, we immediately have the following lemma:

Lemma 2.5. Let $(\bar{\mu}, \bar{\lambda})$ be an optimal solution of (2.11). Then

$$\max_{\lambda \in \mathbf{R}^n} \bar{\phi}(\bar{\mathbf{x}}(\lambda); \lambda) = g(\bar{\mu}, \bar{\lambda}) = \bar{z}.$$

Theorem 2.6. Let $(\mathbf{x}(\lambda), \mathbf{y}(\lambda))$ denote an optimal solution of $[L(\lambda)]$ and let $z(\lambda) = \phi(\mathbf{x}(\lambda), \mathbf{y}(\lambda); \lambda)$. Then

$$\bar{z} \leq z(\bar{\lambda}) \leq z^k,$$

where the first inequality holds strictly if f_i is strictly concave on $I_i^k = [l_i^k, u_i^k]$ and $y_i(\bar{\lambda}) \in \text{int} I_i^k$ for some $i \in M$.

Proof: It follows from Lemma 2.5 that

$$\bar{\lambda} \in \arg \max_{\lambda \in \mathbf{R}^n} \bar{\phi}(\bar{\mathbf{x}}(\lambda); \lambda).$$

Since (2.8) with $\lambda = \bar{\lambda}$ is a relaxation of $[L(\bar{\lambda})]$, we have

$$z(\bar{\lambda}) = \phi(\mathbf{x}(\bar{\lambda}), \mathbf{y}(\bar{\lambda}); \bar{\lambda}) \geq \bar{\phi}(\bar{\mathbf{x}}(\bar{\lambda}); \bar{\lambda}) = \bar{z}.$$

If $y_i(\bar{\lambda}) \in \text{int} I_i^k$ and f_i is strictly concave on I_i^k for some $i \in M$, we have $f_i(y_i(\bar{\lambda})) > \bar{f}_i(y_i(\bar{\lambda}))$ by definition; hence,

$$\phi(\mathbf{x}(\bar{\lambda}), \mathbf{y}(\bar{\lambda}); \bar{\lambda}) > \bar{\phi}(\mathbf{x}(\bar{\lambda}); \bar{\lambda}) \geq \bar{\phi}(\bar{\mathbf{x}}(\bar{\lambda}); \bar{\lambda}). \quad \square$$

While the bound $z(\bar{\lambda})$ turned out to be tighter than \bar{z} , problem $[L(\bar{\lambda})]$ yielding it is a concave minimization in contrast to $[\bar{P}]$. This means that $[L(\bar{\lambda})]$ can have multiple local minima, many of which fail to be global ones. Therefore, as does $[P^k]$, the relaxed problem $[L(\bar{\lambda})]$ itself belongs to the class of multiextremal global optimization which is known to be hard to solve in general [9, 10]. In the next section, however, we will show that the global minimum $z(\lambda)$ of $[L(\lambda)]$ can be computed in $O(mn \log n)$ arithmetic operations and in $O(mn)$ evaluations of f_i s for a given λ .

3. The Solution

Let us again look at problem $[L(\lambda)]$ in detail. We then see that it can be decomposed into m minimization problems, each of which is of the form:

$$\left| \begin{array}{ll} \text{minimize} & z_i = \sum_{j \in N} c_{ij}(\lambda) x_{ij} + f_i(y_i) \\ \text{subject to} & \sum_{j \in N} x_{ij} = y_i, \quad i \in M \\ & 0 \leq x_{ij} \leq b_j, \quad j \in N, \quad y_i \in I_i^k. \end{array} \right. \quad (3.1)$$

Even though the objective function is concave, (3.1) turns into a continuous linear knapsack problem once we fix the value of y_i in the interval $I_i^k = [l_i^k, u_i^k]$. Suppose that the variables x_{ij} , $j \in N$, are arranged in the order

$$c_{ij_1}(\boldsymbol{\lambda}) \leq \cdots \leq c_{ij_p}(\boldsymbol{\lambda}) \leq 0 \leq c_{ij_{p+1}}(\boldsymbol{\lambda}) \leq \cdots \leq c_{ij_n}(\boldsymbol{\lambda}). \quad (3.2)$$

Then an optimal solution of (3.1) with a fixed y_i is given by

$$\tilde{x}_{ij_h}(y_i) = \begin{cases} b_{j_h}, & h = 1, \dots, q-1 \\ y_i - \sum_{h=1}^{q-1} b_{j_h}, & h = q \\ 0, & h = q+1, \dots, n, \end{cases} \quad (3.3)$$

if there is an index $q \leq p$ such that $\sum_{h=1}^{q-1} b_{j_h} \leq y_i < \sum_{h=1}^q b_{j_h}$; otherwise,

$$\tilde{x}_{ij_h}(y_i) = \begin{cases} b_{j_h}, & h = 1, \dots, p \\ 0, & h = p+1, \dots, n. \end{cases} \quad (3.4)$$

Let us denote the value of $(\tilde{x}_{ij}(y_i) \mid j \in N)$ by

$$F_i(y_i) = \sum_{j \in N} c_{ij}(\boldsymbol{\lambda}) \tilde{x}_{ij}(y_i) + f_i(y_i); \quad (3.5)$$

and define

$$\eta_0 = 0; \quad \eta_h = \eta_{h-1} + b_{j_h}, \quad h = 1, \dots, n. \quad (3.6)$$

Lemma 3.1. *The function F_i is concave on the interval $[\eta_{h-1}, \eta_h]$ for each $h = 1, \dots, n$.*

Proof: We see from (3.2) – (3.6) that F_i is composed of a concave function f_i and n piecewise affine functions $c_{ij}(\boldsymbol{\lambda}) \tilde{x}_{ij}(y_i)$, $j \in N$, each of which has at most one break point in $\{\eta_h \mid h = 0, 1, \dots, n\}$. Since a sum of concave and affine functions is concave (see e.g. [3]), the function F_i is concave on each affine piece of $\sum_{j \in N} c_{ij}(\boldsymbol{\lambda}) \tilde{x}_{ij}(y_i)$. \square

Lemma 3.1 guarantees that F_i is minimized at some end point of $[\eta_h, \eta_{h-1}]$ s over the whole interval $[\eta_0, \eta_n] = [0, B]$. Therefore, the optimal value of (3.1) is given by

$$z_i(\boldsymbol{\lambda}) = \min \left\{ F_i(y_i) \mid y_i \in \left(I_i^k \cap \{\eta_h \mid h = 0, 1, \dots, n\} \right) \cup \{l_i^k, u_i^k\} \right\}; \quad (3.7)$$

and the optimal value of $[L(\boldsymbol{\lambda})]$ is

$$z(\boldsymbol{\lambda}) = \sum_{i \in M} z_i(\boldsymbol{\lambda}) + \sum_{j \in N} \lambda_j b_j.$$

Theorem 3.2. *Given $\boldsymbol{\lambda} \geq \mathbf{0}$, the lower bound $z(\boldsymbol{\lambda})$ can be computed in $O(mn \log n)$ arithmetic operations and $O(mn)$ evaluations of f_i s.*

Proof: For each $i \in M$, sorting $c_{ij}(\boldsymbol{\lambda})$ in the order (3.2) requires $O(n \log n)$ arithmetic operations; and (3.7) requires $O(n)$ evaluations of f_i . Their total numbers are $O(mn \log n)$ and $O(mn)$, respectively. \square

The polynomial-time solvability of the concave minimization problem $[L(\boldsymbol{\lambda})]$ is totally due to the *rank-two monotonicity* [12, 19] possessed by the objective function of (3.1). Functions of this class are certainly concave on their domains; but the concavity can be embedded into only a two-dimensional subspace, which enable us to effectively apply parametric programming like the above (see [12] for further details).

3.1. THE BRANCH-AND-BOUND ALGORITHM FOR [PTP]

Using the results obtained so far, we implement the bounding operation through two stages. Let z° denote the least value among feasible solutions of [PTP] in hand. At the first stage, we solve the linear programming relaxed problem $[\bar{P}]$ for a given subproblem $[P^k]$; if the lower bound \bar{z} is less than z° , we set z° to $\min\{z^\circ, f(\bar{\boldsymbol{x}}, \bar{\boldsymbol{y}})\}$ and proceed to the second stage. At the second stage, we solve the Lagrangian relaxed problem $[L(\bar{\boldsymbol{\lambda}})]$. Here, $(\bar{\boldsymbol{\mu}}, \bar{\boldsymbol{\lambda}})$ is an optimal dual solution of $[\bar{P}]$ and hence can be computed in the process of the first stage. Unless the lower bound $z(\bar{\boldsymbol{\lambda}})$ is less than z° , the subproblem $[P^k]$ is fathomed.

If $z(\bar{\boldsymbol{\lambda}}) < z^\circ$, we implement the branching operation in the same way as in Soland [18]. Namely, we choose a factory node $s \in M$ such that

$$s \in \arg \max_{i \in M} \{f_i(\bar{y}_i) - \bar{f}_i(\bar{y}_i)\}, \quad (3.8)$$

and divide the corresponding interval $I_s^k = [l_s^k, u_s^k]$ into two subintervals $I_s^{k_1} = [l_s^k, \bar{y}_s]$ and $I_s^{k_2} = [\bar{y}_s, u_s^k]$. We then update the integral rectangular partition of Y as follows:

$$Y^{k_h} = I_1^k \times \cdots \times I_{s-1}^k \times I_s^{k_h} \times I_{s+1}^k \times \cdots \times I_m^k, \quad h = 1, 2$$

$$\mathcal{Y}' = (\mathcal{Y} \setminus Y^k) \cup \{Y^{k_1}, Y^{k_2}\}.$$

We should note that \bar{y}_s falls in the open interval (l_s^k, u_s^k) . Since we have

$$\bar{z} = \sum_{(i,j) \in A} c_{ij} \bar{x}_{ij} + \sum_{i \in M} \bar{f}_i(\bar{y}_i) < z^\circ \leq f(\bar{\boldsymbol{x}}, \bar{\boldsymbol{y}}) = \sum_{(i,j) \in A} c_{ij} \bar{x}_{ij} + \sum_{i \in M} f_i(\bar{y}_i),$$

at least one of the differences in (3.8) is positive. We see from (2.4) that this can happen only when $\bar{y}_s \in \text{int} I_s^k$. Hence, by the integrality of \bar{y}_i , l_i^k and u_i^k for all $i \in M$, the number of branching operations on each partition set must be finite. Also note that $\bar{\boldsymbol{y}}$ lies in both the partition sets Y^{k_1} and Y^{k_2} newly generated. This implies that $\bar{\boldsymbol{x}}$ is feasible to both $[\bar{P}^{k_1}]$ and $[\bar{P}^{k_2}]$ associated with Y^{k_1} and Y^{k_2} , respectively. We can therefore save the time and memory needed to solve one of them by using $\bar{\boldsymbol{x}}$ as a starting feasible solution if we employ the depth-first-search rule to choose a partition set from \mathcal{Y}' .

The branch-and-bound algorithm for [PTP] is summarized into a recursive form:

algorithm LP.LAGRANGE;

begin

$B := \sum_{j \in N} b_j$;

if $B > \sum_{i \in M} u_i$ **then** [PTP] is infeasible

else

begin

initialize the incumbent $(\mathbf{x}^\circ, \mathbf{y}^\circ) := (\mathbf{0}, \mathbf{0})$ and $z^\circ := +\infty$;

BRANCH/BOUND(Y);

$\mathbf{x}^* := \mathbf{x}^\circ$ and $z^* := z^\circ$

end

end;

procedure BRANCH/BOUND(Y^k);

begin

let $[l_i^k, u_i^k]$, $i \in M$, denote the intervals defining Y^k ;

if $\sum_{i \in M} l_i^k \leq B \leq \sum_{i \in M} u_i^k$ **then**

begin

{The first-stage bounding operation}

let $[P^k]$ denote the subproblem associated with Y^k ;

define the convex envelop \bar{f}_i of f_i over $[l_i^k, u_i^k]$ for each $i \in M$ and construct the linear programming relaxed problem $[\bar{P}]$ of $[P^k]$;

solve $[\bar{P}]$ to obtain its optimal solution $\bar{\mathbf{x}}$, value \bar{z} and dual solution $(\bar{\boldsymbol{\mu}}, \bar{\boldsymbol{\lambda}})$;

if $\bar{z} < z^\circ$ **then**

begin

{The second-stage bounding operation}

$\bar{y}_i := \sum_{j \in N} x_{ij}$ for each $i \in M$;

if $f(\bar{\mathbf{x}}, \bar{\mathbf{y}}) < z^\circ$ **then** update $(\mathbf{x}^\circ, \mathbf{y}^\circ) := (\bar{\mathbf{x}}, \bar{\mathbf{y}})$ and $z^\circ := f(\bar{\mathbf{x}}, \bar{\mathbf{y}})$;

construct the Lagrangian relaxed problem $[L(\bar{\boldsymbol{\lambda}})]$ of $[P^k]$ using $\bar{\boldsymbol{\lambda}}$;

solve $[L(\bar{\boldsymbol{\lambda}})]$ to obtain its optimal value $z(\bar{\boldsymbol{\lambda}})$;

if $z(\bar{\boldsymbol{\lambda}}) < z^\circ$ **then**

begin

{The branching operation}

choose $s \in \arg \max_{i \in M} \{f_i(\bar{y}_i) - \bar{f}_i(\bar{y}_i)\}$;

divide $I_s^k = [l_s^k, u_s^k]$ into $I_s^{k1} = [l_s^k, \bar{y}_s]$ and $I_s^{k2} = [\bar{y}_s, u_s^k]$;

for $h = 1, 2$ **do**

begin

$Y^{kh} := I_1^k \times \dots \times I_{s-1}^k \times I_s^{kh} \times I_{s+1}^k \times \dots \times I_m^k$;

BRANCH/BOUND(Y^{kh})

end

end

end

end

end;

The convex envelopes \bar{f}_i , $i \in M$, can be obtained in $O(m)$ arithmetic operations and $O(m)$ evaluations of f_i s. Problem $[\bar{P}]$ can be constructed in $O(mn)$ arithmetic operations. Therefore, the number of arithmetic operations at the first stage of the bounding operation is dominated by that needed to solve a Hitchcock problem $[\bar{P}]$; it can also be bounded by some lower order polynomial in (m, n) , e.g. $H(m, n) = O((mn \log(m+n))(mn + (m+n) \log(m+n)))$ (see [2]). The second stage, as shown in Theorem 3.2, requires $O(mn \log n)$ arithmetic operations and $O(mn)$ evaluations of f_i s. Consequently, if an evaluation of f_i s can be done in a unit time, the total computational time needed in the procedure BRANCH/BOUND is bounded by $O(H(m, n))$ before its recursive calls.

4. Computational Results

Let us report computational results of testing the algorithm LP_LAGRANGE on randomly generated problems of [PTP].

The algorithm was coded in double precision C language according to the description in Section 3.1. In the procedure BRANCH/BOUND, the Hitchcock problem $[\bar{P}]$ was solved by the stepping-stone algorithm; and $[L(\bar{\lambda})]$ was solved through sorting $c_{ij}(\bar{\lambda})$, $j \in N$, by quicksort for each $i \in M$. Therefore, in our code, the first-stage bounding operation requires pseudo-polynomial time [2]; the second-stage requires $O(mn \log n)$ time on the average but $O(mn^2)$ time in the worst case [1]. In addition to the code LP_LAGRANGE, Soland's algorithm [18] was coded in the same way for the sake of comparison (denoted by SOLAND).

The test problems were generated in the following manner: c_{ij} s were integers drawn from the uniform distribution $[1, 10]$; u_i s were all fixed at 200; b_j s were set to the round-off value of $\alpha(\sum_{i \in M} u_i)/n$ for $\alpha = 0.6, 0.75$ and 0.9 ; and the concave production costs were defined by $f_i(y_i) = \beta\sqrt{y_i}$ for β uniformly random in $[10, 20]$. The size of (m, n) ranged from $(5, 25)$ to $(30, 100)$. For each size, ten instances were solved on a UNIX workstation (hyperSPARC, 150MHz).

Table 4.1 shows the comparison of the codes LP_LAGRANGE and SOLAND on problems of six different sizes. In columns of LP_LAGRANGE, the average number of calls on the procedure BRANCH/BOUND and the average CPU time in seconds (and their maxima in the brackets) are listed for $\alpha = 0.6, 0.75$ and 0.9 ; in columns of SOLAND, the same statistics are listed for $\alpha = 0.75$. We see clearly that LP_LAGRANGE surpasses SOLAND in all respects. There is no doubt that this is caused by the second-stage bounding operation in LP_LAGRANGE, because it is the essential difference between the two codes. More noteworthy is a decrease in the number of calls made by LP_LAGRANGE between $m = 10$ and 15 for each n and α . This tendency is indicated

Table 4.1. Comparison of LP_LAGRANGE and SOLAND.

$m \times n$	LP_LAGRANGE						SOLAND	
	$\alpha = .6$		$\alpha = .75$		$\alpha = .9$		$\alpha = .75$	
	# calls	time	# calls	time	# calls	time	# calls	time
5 × 25	126.5 (239)	.207 (.367)	22.0 (119)	.077 (.183)	1.2 (3)	.042 (.050)	138.0 (225)	.187 (.317)
10 × 25	1244.2 (4439)	3.13 (8.03)	42.4 (143)	.297 (.783)	1.0 (1)	.117 (.133)	8919.8 (31217)	15.23 (53.82)
15 × 25	30.8 (139)	.441 (1.25)	8.8 (19)	.297 (.433)	1.0 (1)	.0.243 (.283)	416949.0 (1660667)	2453.93 (8116.37)
5 × 50	171.4 (243)	1.647 (2.43)	67.6 (141)	1.04 (1.50)	16.8 (41)	.603 (1.08)	392.2 (649)	2.31 (3.78)
10 × 50	3033.0 (13077)	71.46 (239.17)	169.2 (461)	6.85 (11.20)	7.4 (25)	1.07 (1.65)	78374.8 (170401)	1340.97 (3053.12)
15 × 50	1504.8 (4507)	87.68 (260.82)	92.8 (211)	8.42 (16.80)	1.0 (1)	1.48 (1.78)	— (—)	— (—)

* The average time exceeded 10 thousand seconds. When $\alpha = .9$, the number of calls was 88501.6 (291395); the time was 4054.14 (17093.9) seconds.

more clearly by Table 4.2.

Table 4.2 shows the results of LP_LAGRANGE on problems of larger sizes with α fixed at 0.75. For $n = 75$ and 100, the table gives the same statistics as Table 4.1 from $m = 5$ to 30. In either case, we see that the number of calls rises the peak at some m around 10 - 15 and then decreases as m increases. We can therefore expect that LP_LAGRANGE will keep its efficiency up to still larger (m, n) at least for randomly generated problems of [PTP].

Table 4.2. Computational results of LP_LAGRANGE when $\alpha = .75$.

m	$n = 75$				$n = 100$			
	# calls	time	# calls	time	# calls	time	# calls	time
5	82.6 (135)	6.20 (10.38)	110.4 (227)	19.25 (30.48)				
10	433.2 (885)	55.41 (115.10)	1530.6 (6539)	334.64 (1447.67)				
15	711.8 (2309)	130.84 (395.43)	197.2 (515)	122.21 (273.50)				
20	5.2 (21)	11.85 (17.32)	194.2 (1237)	134.98 (657.88)				
25	3.0 (11)	12.76 (16.85)	71.6 (179)	90.78 (175.35)				
30	4.6 (13)	16.36 (22.72)	8.2 (45)	46.33 (89.05)				

5. Concluding Remark

As we have demonstrated in the preceding section, the Lagrangian relaxation $[L(\bar{\lambda})]$ provides a fairly strong lower bound $z(\bar{\lambda})$ for the value z^k of each subproblem $[P^k]$. Before closing the paper, we will show that the bound $z(\bar{\lambda})$ can further be tightened.

Since the total production at factories cannot be below the total demand at warehouses, any feasible solution (\mathbf{x}, \mathbf{y}) of the production-transportation problem (1.1) satisfies

$$\sum_{i \in M} y_i \geq B = \sum_{j \in N} b_j. \quad (5.1)$$

Similarly, if $[P^k]$ has a feasible solution (\mathbf{x}, \mathbf{y}) , it must satisfy (5.1). Therefore, the set of optimal solutions does not change even if we add the constraint (5.1) to $[P^k]$. The resulting Lagrangian relaxation with respect to $\sum_{i \in M} x_{ij} = b_j, j \in M$, is written as

$$[L'(\lambda)] \left\{ \begin{array}{l} \text{minimize} \quad z = \phi(\mathbf{x}, \mathbf{y}; \lambda) \\ \text{subject to} \quad \sum_{i \in M} y_i \geq B \\ \quad \quad \quad \sum_{j \in N} x_{ij} = y_i, \quad i \in M \\ \quad \quad \quad 0 \leq x_{ij} \leq b_j, \quad (i, j) \in A, \quad \mathbf{y} \in Y^k. \end{array} \right.$$

The feasible set of $[L'(\lambda)]$ is obviously included in that of $[L(\lambda)]$. This, together with Theorem 2.6, leads to the following:

Theorem 5.1. Let $(\mathbf{x}'(\lambda), \mathbf{y}'(\lambda))$ denote an optimal solution of $[L'(\lambda)]$ and let $z'(\lambda) = \phi(\mathbf{x}'(\lambda), \mathbf{y}'(\lambda); \lambda)$. Then

$$\bar{z} \leq z(\bar{\lambda}) \leq z'(\bar{\lambda}) \leq z^k.$$

Due to the first constraint, $[L'(\lambda)]$ cannot be decomposed into rank-two monotonic problems like (3.1). However, we can transform it to a specially structured production-transportation problem, for which a pseudo-polynomial algorithm is available [13]. Let us make m copies of the set N , i.e. $N_i = N$ for each $i \in M$; and set $c_{0j}(\lambda)$ to zero for each $j \in \cup_{i \in M} N_i$. Introducing variables $x_{0j}, j \in \cup_{i \in M} N_i$, and $w_i, i \in M$, we have a problem equivalent to $[L'(\lambda)]$:

$$\left\{ \begin{array}{l} \text{minimize} \quad \sum_{i \in M \cup \{0\}} \sum_{j \in N_i} c_{ij}(\lambda) x_{ij} + \sum_{i \in N} f_i(y_i) + \sum_{j \in N} \lambda_j b_j \\ \text{subject to} \quad \sum_{i \in M} w_i \leq (m-1)B \\ \quad \quad \quad \sum_{j \in N_i} x_{0j} = w_i, \quad \sum_{j \in N_i} x_{ij} = y_i, \quad i \in M \\ \quad \quad \quad x_{0j} + x_{ij} = b_j, \quad j \in N_i, \quad i \in M \\ \quad \quad \quad x_{0j}, x_{ij} \geq 0, \quad j \in N_i, \quad i \in M \\ \quad \quad \quad y_i, w_i \geq 0, \quad i \in M. \end{array} \right. \quad (5.2)$$

It is easy to check that the value of (5.2) is equal to $z'(\lambda)$. This problem can be thought of as a production-transportation problem, in which the set of mn warehouses is partitioned into m subsets N_i , $i \in M$; each factory $i \in M$ is allowed to supply only warehouses in its assigned subset N_i ; and factory 0 supplies warehouses in N_i for each $i \in M$ with a total of w_i units. Problems with such a structure can be solved in $O(m^2nB)$ arithmetic operations and $O(mnB)$ evaluations of f_i s if we apply an algorithm proposed in [13]. Although the algorithm is pseudo-polynomial, it will be an effective procedure in the branch-and-bound algorithm for solving [PTP] when n is beyond a hundred and B is relatively small. Computational experiments are now under way, the results of which will be reported elsewhere.

References

- [1] Aho, A.V., J.E. Hopcroft and J.D. Ullman, *Data Structures and Algorithms*, Addison-Wesley (M.A., 1983).
- [2] Ahuja, R.K., T.L. Magnanti and J.B. Orlin, *Network flows: Theory, Algorithms and Applications*, Prentice Hall (N.J., 1993).
- [3] Bazaraa, M.S., H.D. Sherali and C.M. Shetty, *Nonlinear Programming: Theory and Algorithms*, 2nd ed. John Wiley and Sons (N.Y., 1993).
- [4] Falk, J.E. and R.M. Soland, "An algorithm for separable nonconvex programming problems", *Management Science* **15** (1969), 550 – 569.
- [5] Gallo, G., C. Sandi and C. Sodini, "An algorithm for the min concave cost flow problem", *European Journal of Operational Research* **4** (1980), 248 – 255.
- [6] Guisewite, G.M., "Network problems", in R. Horst and P.M. Pardalos (eds.), *Handbook of Global Optimization*, Kluwer Academic Publishers (Dordrecht, 1995).
- [7] Guisewite, G.M. and P.M. Pardalos, "Global search algorithms for minimum concave-cost network flow problems", *Journal of Global Optimization* **1** (1991), 309 – 330.
- [8] Guisewite, G.M. and P.M. Pardalos, "A polynomial time solvable concave network flow problem", *Networks* **23** (1993), 143 – 149.
- [9] Horst, R., P.M. Pardalos and N.V. Thoai, *Introduction to Global Optimization*, Kluwer Academic Publishers (Dordrecht, 1995).
- [10] Horst, R. and H. Tuy, *Global Optimization: Deterministic Approaches*, 2nd ed. Springer-Verlag (Berlin, 1993).
- [11] Klinz, B. and H. Tuy, "Minimum concave-cost network flow problems with a single nonlinear arc cost", in Dungzhu Du and P.M. Pardalos (eds.), *Network Optimization Problems*, World Scientific (Singapore, 1993), 125 – 143.
- [12] Konno, H., P.T. Thach and H. Tuy, *Optimization on Low Rank Nonconvex Structures*, Kluwer Academic Publishers (Dordrecht, 1997).
- [13] Kuno, T. and T. Utsunomiya, "A decomposition algorithm for solving certain classes of production-transportation problems with concave production cost", *Jour-*

- nal of Global Optimization* **8** (1996), 67 – 80.
- [14] Kuno, T. and T. Utsunomiya, “A pseudo-polynomial primal-dual algorithm for globally solving a production-transportation problem”, *Journal of Global Optimization* **11** (1997), 163 – 180.
- [15] Kuno, T., “A pseudo-polynomial algorithm for solving rank three concave production-transportation problems”, *Acta Mathematica Vietnamica* **22** (1997), 159 – 182.
- [16] Magnanti, T.L. and R.T. Wong, “Network design and transportation planning: models and algorithms”, *Transportation Science* **18** (1984), 1 – 55.
- [17] Nemhauser, G.L. and L.A. Wolsey, *Integer and Combinatorial Optimization*, John Wiley and Sons (N.Y., 1988).
- [18] Soland, R.M., “Optimal facility location problems with concave costs”, *Operations Research* **22** (1974), 373 – 382.
- [19] Tuy, H., “Polyhedral annexation, dualization and dimension reduction technique in global optimization”, *Journal of Global Optimization* **1** (1991), 229 – 244.
- [20] Tuy, H., N.D. Dan and S. Ghannadan, “Strongly polynomial time algorithms for certain concave minimization problems on networks”, *Operations Research Letters* **14** (1993), 99 – 109.
- [21] Tuy, H., S. Ghannadan, A. Migdalas and P. Värbrand, “Strongly polynomial algorithm for a production-transportation problem with concave production cost”, *Optimization* **27** (1993), 205 – 227.
- [22] Tuy, H., S. Ghannadan, A. Migdalas and P. Värbrand, “Strongly polynomial algorithm for a production-transportation problem with a fixed number of nonlinear variables”, Technical Report, Department of Mathematics, Linköping University (Linköping, 1993), to appear in *Mathematical Programming*.