Annals of Operations Research 20(1989)67 – 96

THE AUCTION ALGORITHM FOR THE TRANSPORTATION PROBLEM

Dimitri P. BERTSEKAS

Laboratory for Information and Decision Systems, M.I.T., Cambridge, MA 02139, USA

and

David A. CASTANON

Alphatech, Inc., 111 Middlesex Turnpike, Burlington, MA 01803, USA

Abstract

The auction algorithm is a parallel relaxation method for solving the classical assignment problem. It resembles a competitive bidding process whereby unassigned persons bid simultaneously for objects, thereby raising their prices. Once all bids are in, objects are awarded to the highest bidder. This paper generalizes the auction algorithm to solve linear transportation problems. The idea is to convert the transportation problem into an assignment problem, and then to modify the auction algorithm to exploit the special structure of this problem. Computational results show that this modified version of the auction algorithm is very efficient for certain types of transportation problems.

1. Introduction

In this paper, we propose a new relaxation algorithm for linear transportation problems. The algorithm resembles classical coordinate descent, Gauss–Seidel, and Jacobi methods for solving unconstrained nonlinear optimization problems or systems of nonlinear equations. It modifies the dual variables (node prices), either one at a time (Gauss–Seidel version) or all at once (Jacobi version) using only local node information, while aiming to improve the dual cost. It is well suited for implementation on massively parallel machines.

The first relaxation algorithm for linear network flow problem was the auction algorithm for the classical assignment problem, proposed by the first author in 1979 [3] and further discussed in [8,13]. The algorithm operates like an auction whereby unassigned persons bid simultaneously for objects, thereby raising their prices. Once all bids are in, objects are awarded to the highest bidder. The algorithm can also be interpreted as a Jacobi-like relaxation method for solving a dual problem. The variables of the dual problem may be viewed as the prices of the objects and are adjusted upwards as the algorithm progresses. Just as in a real auction, a person's bid is required to be higher than the current price of the object and this provides the mechanism for increasing the object prices. The algorithm makes gradual progress towards a full assignment of persons to objects as the prices of some of the assigned objects becomes sufficiently high, so that unassigned objects become attractive and receive bids.

Computational results [10] show that for large sparse problems, the auction algorithm is superior to the best existing assignment methods even without the benefit of parallelism. The reason for this can be traced to the complexity estimate $O(NA \log(NC))$ for an efficient implementation of the auction algorithm derived in [8,10]; here, N is the number of persons, A is the number of arcs, and C is the maximum absolute value of arc cost coefficient. Competing methods, [1,2,4,14,15, 18–20,23,24], including the Hungarian method, have complexity $O(N^3)$, so for large sparse problems the complexity of the auction algorithm is superior.

This paper extends the auction algorithm to solve linear transportation problems. The basic idea is to convert the transportation problem into an assignment problem by creating multiple copies of persons (or objects) for each source (or sink, respectively), and then to modify the auction algorithm to take advantage of the presence of the multiple copies. Section 2 describes the basic form of the auction algorithm. Section 3 considers a variation of the auction algorithm that takes into account "similar" objects. (Roughly, two objects are called similar if every person to whom they can be assigned considers them as equally valuable.) We also consider a variation of the algorithm that takes into account "similar" persons. (Roughly, two persons are called similar if each person assigns the same value to every object as the other person.) The variation of the auction algorithm that takes into account similar objects is useful, among other things, for handling asymmetric assignment problems, where there are M persons and N objects with $M > N$. We can convert such problems to assignment problems with an equal number of persons and objects by introducing $M - N$ additional similar objects, each offering equal value (e.g. zero) to all persons. The auction algorithm that takes into account both similar persons and similar objects can be restructured so that it solves efficiently transportation problems. This is described in sections 4 and 5, and computational results showing the effectiveness of the corresponding transportation algorithm are given in section 6.

2. The auction algorithm for the assignment problem

Consider N persons who wish to divide among themselves N objects. We number persons and objects as $1, 2, \ldots, N$. For each person i there is a nonempty subset $A(i)$ of objects that can be assigned to i. An *assignment* S is a (possibly empty) set of person-object pairs (i, j) such that:

(a) $j \in A(i)$ for all $(i, j) \in S$.

(b) For each person i there is at most one pair $(i, j) \in S$.

(c) For each object j there is at most one pair $(i, j) \in S$.

A *complete assignment* is an assignment containing N pairs (i.e. every person is assigned to a distinct object). In the context of a given assignment S, we say that person i is *assigned* if there exists an object j such that $(i, j) \in S$; otherwise we say that i is *unassigned.* We use similar terminology for objects. There is a given integer value a_{ij} that a person i associates with an object $j \in A(i)$. We want to find a complete assignment that maximizes

$$\sum_{(i,j) \in S} a_{ij}$$

over all complete assignments S. We call this the *primal assignment problem.* The auction algorithm solves the dual assignment problem ([13,21,25,26])

$$\text{minimize} \quad \sum_{i=1}^{N} r_i + \sum_{j=1}^{N} p_j$$

$$\text{subject to} \quad r_i + p_j \geqslant a_{ij}, \quad \forall i, \text{ and } j \in A(i). \tag{1}$$

The dual variable p_j is called the *price* of j. We call the vector p with coordinates p_j, $j = 1, \ldots, N$, a *price vector.* For a given price vector p, the cost of this problem is minimized when r_i equals the maximum value of $a_{ij} - p_j$ over $j \in A(i)$. We may therefore view the prices p_j as the only optimization variables of the dual problem.

For a given price vector, we define the *value of an object $j \in A(i)$ for a person i* by

$$v_{ij} = a_{ij} - p_j\,. \tag{2}$$

The *profit π_i of person i* is the maximum value of objects $j \in A(i)$, i.e.

$$\pi_i = \max_{j \in A(i)} v_{ij}\,. \tag{3}$$

From linear programming theory, we know that a complete assignment $S = \{(i, j) \mid i = 1, \ldots, N\}$ and a price vector p are simultaneously primal and dual optimal, respectively, if and only if

$$\pi_i = \max_{k \in A(i)} \{a_{ik} - p_k\} = a_{ij} - p_j\,, \qquad \text{for each } (i, j) \in S\,,$$

that is, if and only if each person realizes his profit by being assigned to an object offering maximum value. This is known as the *complementary slackness condition.*

A relaxation of the complementary slackness condition is to allow persons to be assigned to objects that come within ϵ of attaining the maximum in the profit definition (3). Formally, we say that an assignment S (not necessarily complete) and a price vector p satisfy *ϵ-complementary slackness* (ϵ-CS) if

$$\pi_i - \epsilon = \max_{k \in A(i)} \{a_{ik} - p_k\} - \epsilon \leqslant a_{ij} - p_j\,, \qquad \text{for each } (i,j) \in S, \tag{4}$$

where ϵ is a nonnegative constant. The main fact for our purposes is that *a complete assignment S that satisfies ϵ-CS together with some price vector is optimal if $\epsilon < 1/N$.* This was shown in the original proposal of the auction algorithm [3], and is a special case of proposition 4 to be proved later.

We now describe formally the auction algorithm. We fix $\epsilon > 0$, and we start with some (possibly empty) assignment and price vector satisfying ϵ-CS. The algorithm proceeds iteratively and terminates when a complete assignment is obtained. At the start of the generic iteration, we have an assignment S and a price vector p satisfying ϵ-CS. At the end of the iteration, S and some prices are updated while maintaining the ϵ-CS condition. There are two phases in each iteration, the *bidding phase* and the *assignment phase*, described below:

BIDDING PHASE

For each person i that is unassigned under the assigment S:

Compute the current value $v_{ij} = a_{ij} - p_j$ of each object $j \in A(i)$, find a "best" object $j^\star$ having maximum value

$$v_{ij^\star} = \max_{j \in A(i)} v_{ij}\,, \tag{5}$$

and find the best value offered by objects other than $j^\star$

$$w_i = \max_{j \in A(i),\, j \neq j^\star} v_{ij}\,. \tag{6}$$

(If $j^\star$ is the only object in $A(i)$, we define w_i to be $-\infty$ or, for computational purposes, a number that is much smaller than $v_{ij^\star}$.)

Compute the "bid" $b_{ij^\star}$ of person i for object $j^\star$ given by

$$b_{ij^\star} = p_{j^\star} + v_{ij^\star} - w_i + \epsilon = a_{ij^\star} - w_i + \epsilon\,. \tag{7}$$

(We characterize this situation by saying that person i bid for object $j^\star$, and that object $j^\star$ received a bid from person i. The algorithm works if the bid has any value between $p_{j^\star} + \epsilon$ and $p_{j^\star} + v_{ij^\star} - w_i + \epsilon$, but tends to work fastest for the maximal choice (7). The calculation of the bid of a person is illustrated in fig. 1.)

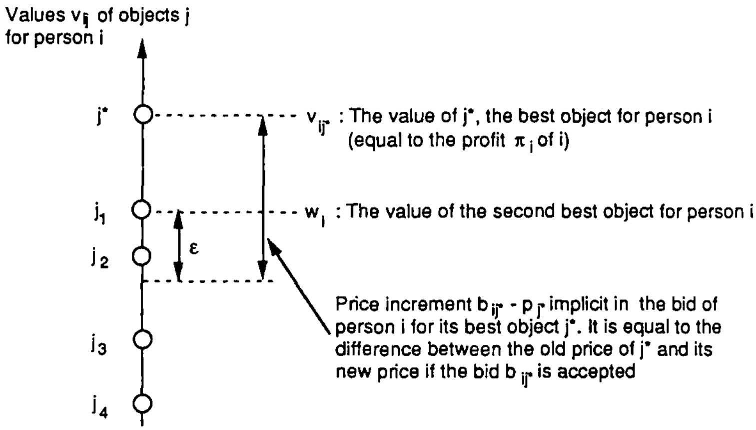

Fig. 1. Illustration of the bid of person i. The objects $j^\star$ and j_1 that offer the best value $v_{ij^\star}$ and second best value w_i, respectively, are determined. The bidding increment $b_{ij^\star} - p_{j^\star}$ is then set to the difference $v_{ij^\star} - w_i$ plus ϵ.

ASSIGNMENT PHASE

For each object j:

Let $P(j)$ be the set of persons from which j received a bid in the bidding phase of the iteration. If $P(j)$ is nonempty, increase p_j to the highest bid

$$p_j := \max_{i \in P(j)} b_{ij} , \tag{8}$$

remove from the assignment S any pair (i, j) (if one exists), and add to S the pair $(i^\star, j)$, where $i^\star$ is some person in $P(j)$ attaining the maximum above. If $P(j)$ is empty, p_j is left unchanged.

An important fact is that *the algorithm preserves* ϵ-CS *throughout its execution*, i.e. if the assignment and price vector available at the start of an iteration satisfy ϵ-CS, the same is true for the assignment and price vector obtained at the end of the iteration. A proof may be found in [3,8,13], where it is also shown that the algorithm terminates in a finite number of iterations (assuming the problem is feasible, i.e. there exists a complete assignment). As a result, if $\epsilon < 1/N$, then the assignment obtained upon termination is optimal.

3. Variations of the auction algorithm

It is possible to apply the auction algorithm of the previous section to a transportation problem after it has been converted to an assignment problem by replacing each source (sink) with multiple copies of persons (objects). Unfortunately, the performance of the method can be quite poor, as shown in the example of fig. 2. Much

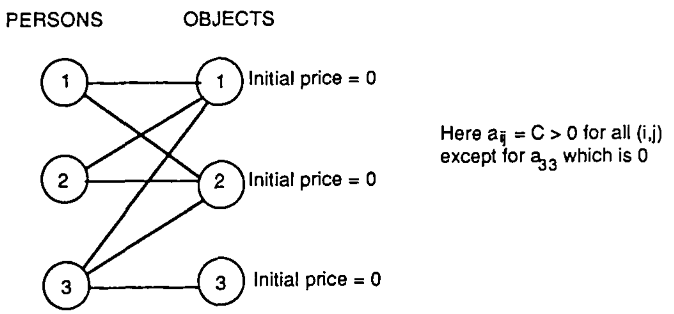

Fig. 2. Example where the number of bidding phases is large and is proportional to C/ϵ. Here, at each bidding phase the persons 1, 2, and 3 bid the prices of objects 1 and 2 up by an increment ϵ until the time that these prices reach the level C. This problem corresponds to a transportation problem, where persons 1 and 2 correspond to a supply node with supply equal to 2 and objects 1 and 2 correspond to a demand node with demand equal to 2.

better performance is obtained with a variation of the auction algorithm which recognizes the special structure derived from the transportation problem. This structure manifests itself in the presence of several "similar" persons and objects, and is formalized below.

Given the assignment problem of the previous section, we say that two objects j and j' are *similar*, and write $j \sim j'$, if

$$\{i \mid j \in A(i)\} = \{i \mid j' \in A(i)\} \tag{9}$$

$$a_{ij} = a_{ij'} \qquad \text{for all } i \text{ with } j \in A(i) . \tag{10}$$

We say that two persons i and i' are *similar*, and write $i \sim i'$, if

$$A(i) = A(i') \tag{11}$$

$$a_{ij} = a_{i'j} \qquad \text{for all } j \in A(i) . \tag{12}$$

For each person (object) i, the set of all persons (objects, respectively) similar to i is called the *similarity class of* i, and is denoted $M(i)$.

For a given price vector p, we define the *price of the similarity class* $M(j)$ *of an object* j as

$$\hat{p}_j = \min_{k \in M(j)} p_k, \quad j = 1, \ldots, N. \tag{13}$$

Note that the profit of a person i given by (3) can also be written as

$$\pi_i = \max_{j \in A(i)} \{a_{ij} - p_j\} = \max_{j \in A(i)} \{a_{ij} - \hat{p}_j\}. \tag{14}$$

It can be seen that:

(a) All persons in the same similarity class have the same profit.

(b) The person profits π_i are determined by the prices $\hat{p}_j$ of the object similarity classes.

It follows that if a complete assignment S and a similarity class price vector $\hat{p}$ satisfy ϵ-CS, and $\epsilon < 1/N$, then S is optimal, even though S and the price vector p may not satisfy ϵ-CS. This is important because in the following algorithms, ϵ-CS of the pair $(S, \hat{p})$ is maintained but ϵ-CS of the pair (S, p) may be violated. An additional benefit of working with the similarity class price vector is that the threshold value for ϵ that guarantees optimality can be increased, as indicated in the following proposition, which will be proved in the next section after we introduce the equivalence between assignment and transportation problems (cf. proposition 4).

PROPOSITION 1

Let

s_p = number of similarity classes of persons,

s_o = number of similarity classes of objects.

If a complete assignment S and a similarity class price vector $\hat{p}$ satisfy ϵ-CS and

$$\epsilon < \frac{1}{\min\{s_p, s_o\}}, \tag{15}$$

then S is optimal.

In what follows in this section, we describe two variations of the auction algorithm. The first variation is actually a special case of the second, but it is easier to understand and illustrates the main ideas more clearly.

THE AUCTION ALGORITHM FOR SIMILAR OBJECTS

Consider a variation of the auction algorithm which is the same as the one of the previous section except that the bidding increments are determined by the values of the similarity classes of the objects rather than the values of the objects themselves. Specifically, in the bidding phase, each person i determines the object $j^\star$ that offers maximum value

$$v_{ij^\star} = \max_{j \in A(i)} v_{ij}$$

(cf. (5)), but the "second best level" w_i is defined now as

$$w_i = \max_{j \in A(i), j \notin M(j^\star)} v_{ij} \tag{16}$$

instead of

$$w_i = \max_{j \in A(i), j \neq j^\star} v_{ij} .$$

Roughly, w_i is the "value of the second best similarity class" rather than the value of the second best object. We refer to this algorithm as AUCTION-SO (for Similar Objects) to distinguish it from the auction algorithm of the previous section, which will be referred to as AUCTION. Because we have

$$\max_{j \in A(i), j \notin M(j^\star)} v_{ij} \leqslant \max_{j \in A(i), j \neq j^\star} v_{ij} ,$$

it follows that the bid (cf. (7))

$$b_{ij^\star} = p_{j^\star} + v_{ij^\star} - w_i + \epsilon , \tag{17}$$

with w_i given as in AUCTION-SO (cf. (16)), will be at least as large as the corresponding bid for AUCTION, where w_i is given by (6). As a result, the price changes of the objects in AUCTION-SO are potentially larger than in AUCTION (see fig. 3). The termination of AUCTION-SO is also potentially faster because, with larger price increases, the gap between values of unassigned and assigned objects will be diminishing faster. As an example, the problem of fig. 2 will be solved much faster with AUCTION-SO than with AUCTION (see fig. 4).

The key factor regarding AUCTION-SO is that, assuming the initial assignment S satisfies ϵ-CS together with the initial similarity class price vector $\hat{p}$, that is,

$$\pi_i - \epsilon = \max_{k \in A(i)} \{a_{ik} - \hat{p}_k\} - \epsilon \leqslant a_{ij} - \hat{p}_j , \quad \text{for each } (i, j) \in S, \tag{18}$$

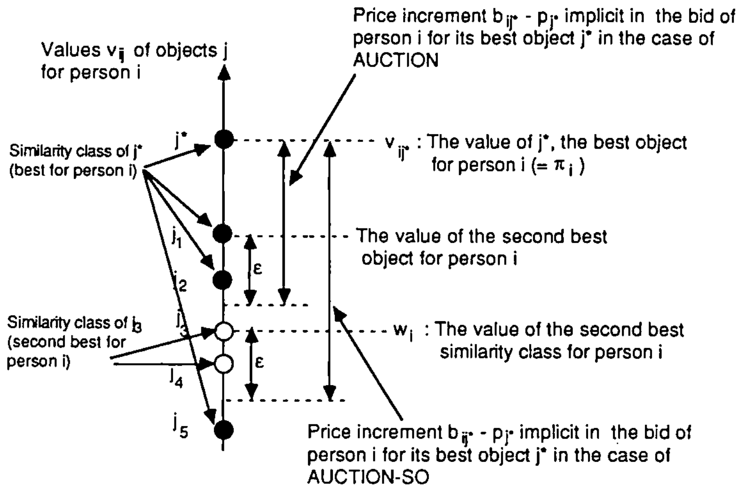

Fig. 3. Illustration of the bid of person i in AUCTION-SO. The object j^* offers the best value v_{ij^*} for person i. When all the second best objects belong to the similarity class of j^* (as in the figure), the bid b_{ij^*} will be higher in AUCTION-SO than in AUCTION.

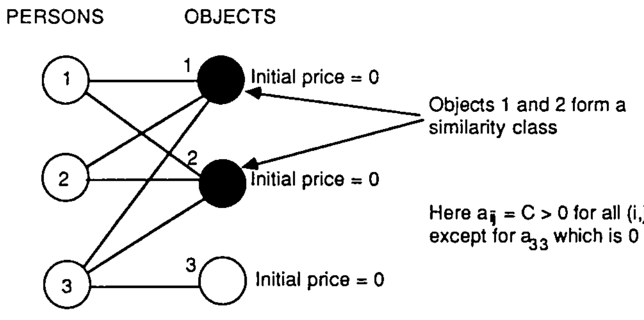

Fig. 4. Application of AUCTION-SO to the problem of fig. 2. Here, the objects 1 and 2 form a similarity class. At the first bidding phase, this similarity class is best for all persons. Person 3 submits a bid $C + \epsilon$ for either object 1 or 2, while persons 1 and 2 submit a "very high" bid for either object 1 or 2 because there is only one similarity class of objects to which they can be assigned. If persons 1 and 2 bid for different objects, then after the first iteration the prices of objects 1 and 2 will be very high and only person 3 will be unassigned. As a result, it is seen that AUCTION-SO will terminate at the next iteration when person 3 will bid for object 3. If persons 1 and 2 bid for the same object at the first iteration, it can be seen that AUCTION-SO will terminate after three iterations.

the same is true of the assignment and the vector $\hat{p}$ obtained at the end of each assignment phase. To show this, we assume that (18) is true at the beginning of all iterations up to a given iteration, and we show that it is true at the end of that iteration. Indeed, suppose that object $j^\star$ received a bid from person i and was assigned to i during the iteration. Then, if p_j and p_j' are the object prices before and after the iteration and $\hat{p}_j'$ is the price of the similarity class of j after the iteration, that is,

$$\hat{p}_j' = \min_{k \in M(j)} p_k' , \qquad j = 1, \ldots, N, \tag{19}$$

we have (cf. (7), (8))

$$p_{j^\star}' = b_{ij^\star} = a_{ij^\star} - w_i + \epsilon . \tag{20}$$

Using (16) and (17) and the easily verifiable fact $p_j' \geqslant p_j$ for all j, it follows that

$$\begin{aligned} a_{ij^\star} - p_{j^\star}' = a_{ij^\star} - b_{ij^\star} = w_i - \epsilon &= \max_{j \in A(i), j \notin M(j^\star)} \{a_{ij} - p_j\} - \epsilon \\ &\geqslant \max_{j \in A(i), j \notin M(j^\star)} \{a_{ij} - p_j'\} - \epsilon = \max_{j \in A(i), j \notin M(j^\star)} \{a_{ij} - \hat{p}_j'\} - \epsilon . \end{aligned} \tag{21}$$

We also have $\hat{p}_{j^\star}' \leqslant p_{j^\star}'$, so we obtain

$$a_{ij^\star} - \hat{p}_{j^\star}' \geqslant \max_{j \in A(i), j \notin M(j^\star)} \{a_{ij} - \hat{p}_j'\} - \epsilon . \tag{22}$$

Since we have $a_{ij^\star} - \hat{p}_{j^\star}' = a_{ij} - \hat{p}_j'$ for all $j \in M(j^\star)$, we see that (22) implies that the ϵ-CS condition (18) holds after the assignment phase of an iteration, for any pair $(i, j^\star)$ that entered the assignment during the iteration. Consider also a pair $(i, j^\star)$ that belonged to the assignment just before an iteration and also belongs to the assignment after the iteration. Let p'' be the price vector just after the iteration in which $(i, j^\star)$ entered the assignment. Then, as in (21), we obtain

$$a_{ij^\star} - p_{j^\star}'' \geqslant \max_{j \in A(i), j \notin M(j^\star)} \{a_{ij} - \hat{p}_j''\} - \epsilon , \tag{23}$$

where

$$\hat{p}_j'' = \min_{k \in M(j)} p_k'' .$$

We have $\hat{p}_j' \geqslant \hat{p}_j''$ since the prices are monotonically nondecreasing, and $p_{j^\star}'' = p_{j^\star}'$ since $j^\star$ must not have received a bid since it was last assigned to i. Therefore, from (23) we obtain

$$a_{ij^\star} - p'_{j^\star} \geqslant \max_{j \in A(i), j \notin M(j^\star)} \{a_{ij} - \hat{p}'_j\} - \epsilon .$$

In view of the fact $\hat{p}'_{j^\star} \leqslant p'_{j^\star}$, we obtain the ϵ-CS condition (18) for the pair $(i, j^\star)$.

The conclusion is that if AUCTION-SO terminates, the assignment obtained at termination is complete and satisfies ϵ-CS together with the corresponding price vector $\hat{p}$. Thus, if

$$\epsilon < \frac{1}{\text{number of object similarity classes}}$$

(cf. proposition 1), the assignment obtained is optimal. There remains to show that AUCTION-SO terminates. We will show this in the context of the following more general algorithm, that takes into account both similar persons and similar objects.

THE AUCTION ALGORITHM FOR SIMILAR PERSONS AND OBJECTS

We consider a variation of the auction algorithm that takes into account similar persons. The idea is to submit a common bid for all persons in a similarity class if at least one person in the class is unassigned. As a result, persons in the same similarity class do not "compete" against each other for the same object, and the bids submitted are higher than they would otherwise be. This idea is combined with the variation discussed earlier that takes into account similar objects to accelerate termination even further.

The algorithm will now be described formally. We fix $\epsilon > 0$, and we start with some assignment S (possibly the empty assignment) and a price vector p satisfying the following condition:

ϵ-COMPLEMENTARY SLACKNESS STRENGTHENED (ϵ-CSS): If $(i, j) \in S$, then

$$a_{ij} - p_j \geqslant \max_{k \in A(i),\, k \notin M(j)} \{a_{ik} - p_k\} - \epsilon , \tag{24}$$

that is, the value of j for i can be worse by at most ϵ over the highest value offered by similarity classes other than the one of j.

We note that the ϵ-CSS condition is stronger than (i.e. implies) ϵ-CS of S and the similarity class price vector $\hat{p}$ (the reverse is not true). Indeed, from (24) and the definition (13) of $\hat{p}$, we have for all $(i, j) \in S$,

$$a_{ij} - \hat{p}_j \geqslant a_{ij} - p_j \geqslant \max_{k \in A(i),\, k \notin M(j)} \{a_{ik} - p_k\} - \epsilon .$$

Since we also have

$$a_{ij} - \hat{p}_j = \max_{k \in M(j)} \{a_{ik} - p_k\} ,$$

the ϵ-CS condition

$$a_{ij} - \hat{p}_j \geqslant \max_{k \in A(i)} \{a_{ik} - p_k\} - \epsilon = \max_{k \in A(i)} \{a_{ik} - \hat{p}_k\} - \epsilon, \ \forall (i, j) \in S$$

follows.

The algorithm proceeds iteratively, and terminates when a complete assignment is obtained. At the start of the generic iteration, we have a pair (S, p) satisfying ϵ-CSS. At the end of the iteration, we obtain another pair (S', p') that will be shown to satisfy ϵ-CSS. As earlier, there are two phases in each iteration, the *bidding phase* and the *assignment phase*, described below:

BIDDING PHASE

For each similarity class of persons $M(i)$ containing a person i that is unassigned under the assignment S:

Compute the current value $v_{ij} = a_{ij} - p_j$ of each object $j \in A(i)$. Let $i_1, i_2, \ldots, i_m$ be the persons in $M(i)$ that are assigned under S, and let $j_1, j_2, \ldots, j_m$ be the corresponding objects to which they are assigned. Let $i_{m+1}, i_{m+2}, \ldots, i_n$ be the persons in $M(i)$ that are unassigned under S. Denote also by $j_{m+1}, j_{m+2}, \ldots, j_{n'}$ the objects that belong to $A(i)$ and are not assigned to any person in $M(i)$ under S, and assume that these objects are ranked in order of decreasing value, i.e.

$$v_{ij_{m+1}} \geqslant v_{ij_{m+2}} \geqslant \ldots \geqslant v_{ij_{n'}} . \tag{25}$$

Compute the scalar w_i (which is analogous to the scalar w_i of (6) and (16)) as follows:

Case (a)

If $n < n'$ and $j_1, j_2, \ldots, j_n$ do not belong to the same similarity class, let

$$w_i = v_{ij_{n+1}} . \tag{26}$$

Case (b)

If $n < n'$ and $j_1, j_2, \ldots, j_n$ belong to the same similarity class, let w_i be the value v_{ij} of the first object $j \in \{j_{n+1}, \ldots, j_{n'}\}$ that does not belong to the common similarity class $M(j_1)$.

Case (c)

If $n = n'$, which is the exceptional case where all the objects in $A(i)$ must be assigned to the persons in the similarity class of person i, we define w_i to be $-\infty$ or, for computational purposes, a number that is much smaller than $\min_{j \in A(i)} v_{ij}$.

Compute the "bid" of each person $i_1, i_2, \ldots, i_n$ for the object $j_1, j_2, \ldots, j_n$, respectively, as

$$b_{i_k j_k} = a_{i_k j_k} - w_i + \epsilon . \tag{27}$$

(As before, we characterize this situation by saying that person i_k bid for object j_k, and that object j_k received a bid from person i_k. Note here that the objects $i_1, \ldots, i_m$, which are assigned under S will bid for their assigned objects $j_1, j_2, \ldots, j_m$. Cases (a) and (b) and the corresponding bids are illustrated in fig. 5.)

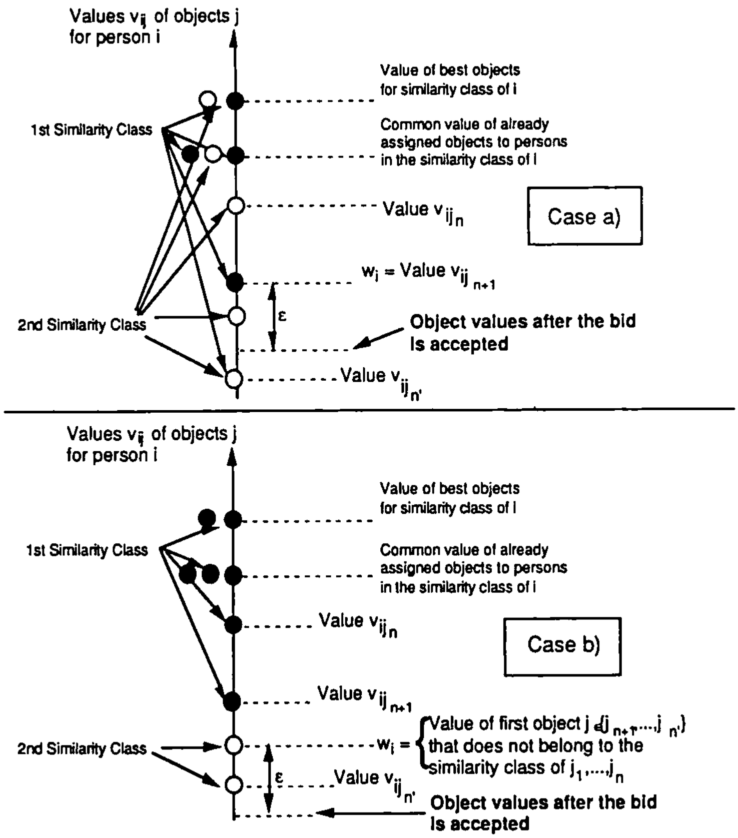

Fig. 5. Illustration of the bid of a similarity class of an unassigned person i. There are n persons in this class. The objects that the similarity class bids for are $j_1, \ldots, j_n$. The common value of those of the objects for which the bid is accepted is shown for case (a) (objects $j_1, \ldots, j_n$ belong to the different similarity classes) and for case (b) (objects $j_1, \ldots, j_n$ belong to the same similarity class).

ASSIGNMENT PHASE

For each object j:

Let $P(j)$ be the set of persons from which j received a bid in the bidding phase of the iteration. If $P(j)$ is nonempty, set p_j to the highest bid, i.e.

$$p_j' = \max_{i \in P(j)} b_{ij}, \tag{28}$$

remove from the assignment S any pair (i, j) (if one exists), and add to S the pair (i^*, j), where i^* is some person in $P(j)$ attaining the maximum above. If $P(j)$ is empty, p_j is left unchanged, i.e. $p_j' = p_j$.

The problem can be easily transformed so that the exceptional case (c) of the bidding phase does not arise. To simplify the subsequent analysis, we will henceforth assume that, if necessary, this transformation is done so that case (c) never arises. Our results, however, hold even when case (c) can arise, provided we allow the object prices to take the value $-\infty$ and we interpret appropriately the arithmetic of extended real numbers.

The preceding algorithm will be referred to as AUCTION-SOP (for Similar Objects and Persons). Note that the case where all similarity classes of persons consist of a single person corresponds to $m = 0$ and $n = 1$ in the bidding phase. Then case (a) of the bidding phase never arises and AUCTION-SOP coincides with AUCTION-SO. Note also that the structure of the algorithm is such that if at the end of an iteration we have $(i, j) \in S$ and $(i', j') \in S$, and $i \sim i'$, then

$$a_{ij} - p_j = a_{i'j'} - p_{j'},$$

that is, objects assigned to persons from the same similarity class have the same value for these persons. If in addition $j \sim j'$, it follows that $p_j = p_{j'}$.

We now show the validity of AUCTION-SOP.

PROPOSITION 2

At each iteration prior to termination of AUCTION-SOP, all the object prices do not decrease, and at least one object price increases by at least ϵ. Furthermore, ϵ-CSS holds at the end of each iteration.

Proof

Suppose ϵ-CSS holds before a given iteration, and let p_j and p_j' be the prices of the objects j before and after the iteration, respectively. Let also π_i be the person profits and S be the assignment before the iteration. Suppose that person $i_k \in M(i)$ bids for object $j_k \in M(j)$ during the iteration. We will show that

$$b_{i_k j_k} \geq p_{j_k} \quad \text{if} \quad (i_k, j_k) \in S \tag{29}$$

$$b_{i_k j_k} \geq p_{j_k} + \epsilon \quad \text{if} \quad (i_k, j_k) \notin S . \tag{30}$$

Suppose first that $(i_k, j_k) \in S$. Then by ϵ-CSS we have

$$a_{i_k j_k} - p_{j_k} \geq \max_{s \in A(i), s \notin M(j)} \{a_{is} - p_s\} - \epsilon . \tag{31}$$

If the bid of the similarity class of i_k is based on case (a) in the bidding phase, we have

$$\max_{s \in A(i), s \notin M(j)} \{a_{is} - p_s\} \geq v_{ij_{n+1}} = w_i ,$$

while if case (b) holds, we have

$$\max_{s \in A(i), s \notin M(j)} \{a_{is} - p_s\} = w_i .$$

Thus, in either case, the ϵ-CS condition (31) yields

$$a_{i_k j_k} - p_{j_k} \geq w_i - \epsilon .$$

Using this relation together with the bid definition (27), we obtain

$$b_{i_k j_k} \geq p_{j_k} ,$$

proving (29).

Suppose next that $(i_k, j_k) \notin S$. Then in both case (a) and (b), in view of the ordering (25), we have

$$a_{i_k j_k} - p_{j_k} \geq w_i ,$$

and using this relation together with (27), we obtain

$$b_{i_k j_k} \geq p_{j_k} + \epsilon ,$$

proving (30).

Since the price of j_k after the iteration is equal to the highest bid (cf. (28)), from (29) and (30) we obtain

$$p'_{j_k} \geq p_{j_k} \quad \text{if} \quad (i_k, j_k) \in S \tag{32}$$

$$p'_{j_k} \geq p_{j_k} + \epsilon \quad \text{if} \quad (i_k, j_k) \notin S . \tag{33}$$

We also have $p_j' = p_j$ for every j that did not receive a bid during the iteration. Thus, the object prices cannot decrease during an iteration. Furthermore, since at least one unassigned person bids at each iteration, it follows from (33) that at least one object price will increase by at least ϵ.

We next show that ϵ-CSS is satisfied following an iteration. Suppose that (i_k, j_k), with $i_k \in M(i)$ and $j_k \in M(j)$, belongs to the assignment following an iteration and that i_k bid for j_k during the iteration. Then

$$\begin{aligned} a_{i_k j_k} - p'_{j_k} &= a_{i_k j_k} - b_{i_k j_k} = w_i - \epsilon \\ &\geqslant \max\{a_{is} - p_s \mid s \in A(i), s \notin M(j_k), s \text{ did not receive a bid from any person in } M(i)\} - \epsilon \\ &\geqslant \max\{a_{is} - p'_s \mid s \in A(i), s \notin M(j_k), s \text{ did not receive a bid from any person in } M(i)\} - \epsilon, \end{aligned}$$

where the next to last inequality holds as an equation if case (b) holds when the bid of the similarity class of i_k is calculated. We also have

$$a_{i_k j_k} - p'_{j_k} = a_{i_m s} - b_{i_m s} \geqslant a_{i_m s} - p'_s$$

for all $s \in A(i)$ such that s received a bid from a person $i_m \in M(i)$. By combining the last two relations, we see that ϵ-CSS holds at the end of the iteration.

Consider now the case where (i_k, j_k), with $i_k \in M(i)$ and $j_k \in M(j)$, belongs to the assignment following an iteration, but i_k did not bid for j_k during the iteration because all persons in $M(i_k)$ were assigned during the iteration. Let p'' be the price vector at the end of the last iteration where all persons in $M(i_k)$ were assigned. Then by ϵ-CSS we have

$$a_{i_k j_k} - p''_{j_k} \geqslant \max_{s \in A(i_k), s \notin M(j_k)} \{a_{i_k s} - p''_s\} - \epsilon .$$

It is seen that the price of j_k remained unchanged since the last iteration where all persons in $M(i_k)$ were assigned, while the other prices could not have decreased, i.e.

$$p''_{j_k} = p'_{j_k}$$

$$p''_s \leqslant p'_s \qquad \forall s \in A(i_k) .$$

By combining the last three relations, we see that ϵ-CSS holds for the pair (i_k, j_k).

Q.E.D.

PROPOSITION 3

AUCTION-SOP terminates if the problem is feasible, i.e. if there exists at least one complete assignment.

Proof

We make the following observations:

(a) Once an object is assigned, it remains assigned throughout the remainder of the algorithm's duration. Furthermore, except at termination, there will always exist at least one object that has never been assigned and that has a price equal to its initial price. This is due to the fact that a bidding and assignment phase can result in a reassignment of an already assigned object to a different person, but cannot result in the object becoming unassigned.

(b) When the similarity class of a person bids during an iteration, the price of at least one of the objects that it bids for increases by at least ϵ (cf. proposition 2).

(c) The profit π_i of a person i decreases by at least ϵ when the person (together with all other persons in its similarity class) bids during a number of iterations which is greater than or equal to

$$\sum_{j \in A(i)} |M(j)| \,, \tag{34}$$

where $|M(j)|$ is the cardinality of the similarity class of object j. The reason is that the number of objects that attain within ϵ the maximum in the definition (3) of π_i is at most equal to the sum (34), and the price of each of these objects must increase (by at least ϵ, thereby decreasing π_i by at least ϵ) before the similarity class of person i will submit a bid for any other objects.

We now argue by contradiction and assume that the algorithm never terminates. Then the prices of a proper and nonempty subset J^∞ of objects increase to $+\infty$ (cf. observations (a) and (b) above), while the profits π_i of a nonempty subset I^∞ of persons decrease to $-\infty$ (cf. observation (c) above). For all $i \in I^\infty$, we must have $J^\infty \supset A(i)$, since otherwise, from definition (3), it is seen that the profit of i would be bounded. The objects in J^∞ after some iteration can only be assigned to objects from I^∞, since the profits of persons not in I^∞ remain bounded and the prices of objects in J^∞ increase to $+\infty$. Furthermore, in view of observation (c) above, only persons from I^∞ will be unassigned after some iteration. Therefore, the cardinality of I^∞ is greater than the cardinality of J^∞, while we have $J^\infty \supset A(i)$ for all i in I^∞. This contradicts the existence of a complete assignment. Q.E.D.

By combining now propositions 1–3, we see that if the problem is feasible and $\epsilon < 1/\min\{s_p, s_o\}$, then AUCTION-SOP will terminate with an optimal assignment.

4. The auction algorithm for the transportation problem

We now consider a transportation problem of the form

$$\text{maximize} \quad \sum_{i=1}^{N} \sum_{j \in A(i)} a_{ij} f_{ij}$$

$$\text{subject to} \quad \sum_{j \in A(i)} f_{ij} = \alpha_i, \quad i = 1, \ldots, N$$

$$\sum_{\{i \mid j \in A(i)\}} f_{ij} = \beta_j, \quad j = 1, \ldots, M$$

$$0 \leqslant f_{ij}, \tag{TP}$$

where a_{ij} and $A(i)$ are as in the assignment problem, and α_i and β_j are given positive integers called the *supply of source* i and the *demand of sink* j, respectively. For feasibility, it is necessary to assume that

$$\sum_{i=1}^{N} \alpha_i = \sum_{j=1}^{M} \beta_j . \tag{35}$$

This problem can be converted into an assignment problem by replacing source i (sink j) with α_i similar persons (or β_j similar objects, respectively). We call this assignment problem (ATP). An integer flow variable f_{ij} in (TP) is equivalent to assignment of f_{ij} similar persons (corresponding to source i) to f_{ij} similar objects (corresponding to sink j). A flow vector $f = \{f_{ij} \mid j \in A(i)\}$ satisfying

$$0 \leqslant f_{ij}, \qquad \text{for all } (i, j) \text{ with } j \in A(i), \tag{36}$$

$$\sum_j f_{ij} \leqslant \alpha_i, \text{ for all } i, \qquad \text{and} \qquad \sum_i f_{ij} \leqslant \beta_j, \text{ for all } j, \tag{37}$$

corresponds to an assignment in (ATP). This assignment is complete if and only if the flow vector f is feasible in (TP).

Consider now the auction algorithm for (ATP) as modified in the previous section to take into account similar persons and objects. We assume that the initial assignment and price vector satisfy the ϵ-CSS condition of the previous section, and that initially all objects in the same similarity class have equal prices. The assignment and price vector pairs generated by the modified auction algorithm will satisfy ϵ-CSS at the beginning of each iteration. Furthermore, all objects from the same similarity class which are assigned to persons from the same similarity class have equal prices;

see the note preceding proposition 2. Therefore, the price vector can be described by specifying, for each (i, j) with $j \in A(i)$, a common price y_{ij} for all objects of the similarity class of j which are assigned to persons in the similarity class of i, together with the common initial price y_{0j} for the unassigned objects in the similarity class of j.

We denote by $R(j)$ the set of indices $i = 1, 2, \ldots, N$ for which there are some objects in the similarity class of j which are assigned to some persons in the similarity class of i, together with the index 0, if there is some unassigned object in the similarity class of j. Because initially all objects in the same similarity class have equal prices, and all prices are nondecreasing, we see that at the beginning of each iteration we have

$$y_{0j} \leqslant y_{ij}, \quad \text{if} \quad 0 \in R(j) \quad \text{and} \quad i \in R(j) . \tag{38}$$

Furthermore, from ϵ-CSS we see that the price vector p with coordinates given by

$$p_j = \min_{i \in R(j)} y_{ij} , \tag{39}$$

satisfies the ϵ-CS condition

$$i \in R(j), \quad i \neq 0 \Rightarrow \pi_i - \epsilon \leqslant a_{ij} - p_j ,$$

where π_i, the profit of source i, is given by (cf. (3))

$$\pi_i = \max_{\{k \mid j \in A(k)\}} \{a_{kj} - p_j\} . \tag{40}$$

In the context of the transportation problem (TP), this condition is restated as

$$f_{ij} > 0 \Rightarrow \pi_i - \epsilon \leqslant a_{ij} - p_j , \tag{41}$$

which may be viewed as a version of the ϵ-CS condition of [6,11,9,10] as applied to the transportation problem (TP).

The following proposition derives the appropriate threshold value for ϵ that guarantees optimality of a feasible flow vector f and a price vector p that satisfy the ϵ-CS condition (41).

PROPOSITION 4

If the feasible flow vector f and the price vector p satisfy the ϵ-CS condition (41) with $\epsilon < 1/\min\{M, N\}$, then f is optimal.

Proof

If f is not optimal, there must exist a cycle with no repeated nodes

$$Y = (i_1, j_2, i_2, j_3, \ldots, i_{k-1}, j_k, i_k, j_1, i_1)$$

along which flow can be pushed without violating the feasibility of f and with an improvement of the primal cost. Here the nodes i_m and j_m are sources and sinks, respectively, and we have $j_m \in A(i_m), j_{m+1} \in A(i_m), m = 1, 2, \ldots, k-1, j_k \in A(i_k)$, $j_1 \in A(i_k)$. Because Y has no repeated nodes, we have $k \leqslant \min\{M, N\}$, which based on the hypothesis on ϵ, implies that

$$k\epsilon < 1 \,. \tag{42}$$

Furthermore, we must have

$$f_{i_m j_m} > 1, \quad m = 1, \ldots, k \tag{43}$$

(in order to be able to push flow from j_m back to i_m), and

$$\sum_{m=1}^{k} a_{i_m j_m} + 1 \leqslant a_{i_k j_1} + \sum_{m=2}^{k} a_{i_{m-1} j_m} \tag{44}$$

(since pushing flow along Y improves the cost and the coefficients a_{ij} are integer). It follows that

$$\begin{aligned} \sum_{m=1}^{k} (a_{i_m j_m} - p_{j_m}) + 1 &\leqslant (a_{i_k j_1} - p_{j_1}) \\ &\quad + \sum_{m=2}^{k} (a_{i_{m-1} j_m} - p_{j_{m-1}}) \leqslant \sum_{m=1}^{k} \pi_{i_m} \,. \end{aligned} \tag{45}$$

Using (43) and the ϵ-CS condition (41) we obtain

$$\pi_{i_m} - \epsilon \leqslant a_{i_m j_m} - p_{j_m} \,, \quad m = 1, \ldots, k.. \tag{46}$$

The last two inequalities and the condition $k\epsilon < 1$ (cf. (42)) yield

$$\begin{aligned} \sum_{m=1}^{k} (a_{i_m j_m} - p_{j_m}) + 1 &\leqslant \sum_{m=1}^{k} \pi_{i_m} \\ &\leqslant \sum_{m=1}^{k} (a_{i_m j_m} - p_{j_m}) + k\epsilon < \sum_{m=1}^{k} (a_{i_m j_m} - p_{j_m}) + 1 \,, \end{aligned}$$

which is a contradiction. Q.E.D.

Based on the equivalence of transportation problems, and assignment problems with similar persons and objects discussed earlier, it is seen that proposition 1 is a special case of proposition 4.

We now describe our transportation algorithm by restating AUCTION-SOP for the equivalent assignment problem (ATP) in terms of a flow variable f_{ij} and a price variable for each (i, j) with $j \in A(i)$, together with a set of initial price variables y_{0j} for all sinks j. The generic iteration of this algorithm consists of a bidding phase and an assignment phase, stated below. At the start of the iteration we have a set of flow variables f_{ij} and price variables y_{ij} satisfying conditions (39), (40) and (41). At the end of the iteration we obtain a set of flow variables f'_{ij} and price variables y'_{ij} satisfying the same conditions. To simplify the statement of the algorithm we define, for any flow vector f,

$$f_{0j} = \beta_j - \sum_{\{i \mid j \in A(i)\}} f_{ij}, \qquad j = 1, 2, \ldots, M. \tag{47}$$

We also assume that

$$\alpha_i < \sum_{j \in A(i)} \beta_j ,$$

for all i; this guarantees that case (c) in the bidding phase of AUCTION-SOP does not arise.

BIDDING PHASE

For each source i such that $\sum_j f_{ij} < \alpha_i$:

Consider the collection

$$\Pi(i) = \{a_{ij} - y_{kj} \mid j \in A(i), \text{ and either } k = 0 \text{ and } f_{0j} > 0 \text{ or}$$
$$k \neq i \text{ and } j \in A(k), f_{kj} > 0\}, \tag{48}$$

and assume that $\Pi(i)$ is ordered in decreasing order, i.e. for some $\bar{n}$ we have

$$\Pi(i) = \left\{a_{ij_1} - y_{k_1 j_1}, a_{ij_2} - y_{k_2 j_2}, \ldots, a_{ij_n} - y_{k_{\bar{n}} j_{\bar{n}}}\right\},$$

with

$$a_{ij_n} - y_{k_n j_n} \geq a_{ij_{n+1}} - y_{k_{n+1} j_{n+1}}, \qquad \text{for all } n = 1, \ldots, \bar{n} - 1.$$

Let m be the smallest integer m' such that

$$f_{k_1 j_1} + \ldots + f_{k_{m'} j_{m'}} > \alpha_i - \sum_{j \in A(i)} f_{ij} .$$

Define flows $\hat{f}_{ij}$ for all $j \in A(i), j \neq j_m$, given by

$$\hat{f}_{ij} = f_{ij} \quad \text{if} \quad j \neq j_1, \ldots, j_m$$

$$\hat{f}_{ij} = f_{ij} + \sum_{\{j_n | j = j_n, n = 1, \ldots, m-1\}} f_{k_n j_n} \quad \text{if} \quad j \in \{j_1, j_2, \ldots, j_{m-1}\}$$

and then define

$$\hat{f}_{ij_m} = \alpha_i - \sum_{\{j \in A(i) | j \neq j_m\}} \hat{f}_{ij} .$$

Compute the scalar w_i as follows: If $\hat{f}_{ij} > 0$ for more than one sink j, then

$$w_i = a_{ij} - (a_{ij_m} - y_{k_m j_m}) ,$$

and otherwise

$$w_i = a_{ij} - (a_{ij_n} - y_{k_n j_n}) ,$$

where n is the first integer n' for which $j_{n'} \neq j_1$.

Compute the "bid" of source i for each flow $\hat{f}_{ij} > 0$ as

$$b_{ij} = a_{ij} - w_i + \epsilon .$$

(As before, we characterize this situation by saying that source i bid for a flow increment $\hat{f}_{ij}$ of sink j at a price b_{ij}, and that sink j received a bid from source i for a flow increment $\hat{f}_{ij}$ at a price b_{ij}.)

ASSIGNMENT PHASE

For each sink j:

Let $P(j)$ be the set of sources from which j received a bid for a positive flow increment in the bidding phase of the iteration. Assume that $P(j)$ is ordered in decreasing bid value, i.e. $P(j)$ is of the form

$$P(j) = \{i_1, i_2, \ldots, i_{\overline{m}}\} ,$$

where

$$\hat{f}_{i_m j} > 0, \ \ b_{i_m j} \geqslant b_{i_{m+1} j}, \quad \text{for } m = 1, \ldots, \overline{m} - 1.$$

Let

$$n = \bar{m} \quad \text{if} \quad \sum_{m=1}^{\bar{m}} \hat{f}_{i_m j} \leqslant \beta_j$$

and, otherwise, let n be the smallest integer m' such that

$$\sum_{m=1}^{m'} \hat{f}_{i_m j} > \beta_j .$$

Then update the flows f_{ij}, for i such that $j \in A(i)$, by

$$f'_{ij} = \hat{f}_{ij} \qquad \text{if } i \in \{i_1, \ldots, i_{n-1}\}$$

$$f'_{ij} = \hat{f}_{i_n j} - \max\left\{0, \sum_{m=1}^{n} \hat{f}_{i_m j} - \beta_j\right\} \qquad \text{if } i = i_n$$

$$f'_{ij} = 0 \qquad \text{otherwise} .$$

Set also

$$y'_{ij} = b_{ij} , \qquad \text{for } i \in \{i_1, \ldots, i_n\}$$

$$y'_{0j} = y_{0j} .$$

Based on the results proved for the auction algorithm of the previous section, the transportation algorithm above terminates with an optimal solution provided that the transportation problem (TP) is feasible and $\epsilon < 1/\min\{M, N\}$.

5. The auction algorithm for inequality constrained transportation problems

The ideas and algorithms of the previous sections can be extended to inequality constrained assignment and transportation problems of the form

$$\begin{aligned} &\text{maximize} \quad \sum_{i=1}^{N} \sum_{j \in A(i)} a_{ij} f_{ij} \\ &\text{subject to} \quad \sum_{j \in A(i)} f_{ij} \leqslant \alpha_i, \quad i = 1, \ldots, N \\ &\qquad \sum_{\{i | j \in A(i)\}} f_{ij} \leqslant \beta_j, \quad j = 1, \ldots, M \\ &\qquad 0 \leqslant f_{ij} . \end{aligned} \tag{TPI}$$

The ϵ-complementary slackness conditions take the form

$$f_{ij} > 0 \qquad \Rightarrow \pi_i - \epsilon \leqslant a_{ij} - p_j \,, \tag{49}$$

$$\pi_i \geqslant 0, \qquad \forall i = 1, \ldots, N$$

$$p_j \geqslant 0, \qquad \forall j = 1, \ldots, M$$

$$\sum_{\{i | j \in A(i)\}} f_{ij} < \beta_j \Rightarrow p_j = 0, \ \forall j = 1, \ldots, M \,,$$

where the source profits π_i are given by

$$\pi_i = \max_{\{k | j \in A(k)\}} \{a_{kj} - p_j\}, \qquad \forall j = 1, \ldots, N \,.$$

It can be shown that if a feasible flow vector f satisfies, together with a price vector p, the above conditions, then f is optimal if $\epsilon < 1/\min\{M, N\}$.

Finally, the auction algorithm of the previous section can be used to solve inequality constrained problems, provided the initial flow and price vector pair satisfies the above ϵ-CS conditions and the set $\Pi(i)$ of eq. (48) in the bidding phase is modified to include only scalars $a_{ij} - y_{kj}$ that are nonnegative. In particular, if due to this restriction the set $\Pi(i)$ is empty, then source i does not participate in the bidding phase. The algorithm terminates when for all sources i either the set $\Pi(i)$ is empty or the supply α_i is assigned, that is,

$$\sum_{j \in A(i)} f_{ij} = \alpha_i \,.$$

6. Computational results

The algorithm of section 4 for (equality constrained) transportation problems was implemented in a code called TRANSAUCTION, and was compared with the following state-of-the-art codes:

(1) AUCTION (written by Bertsekas [8]): This is a public domain code implementing the auction algorithm for the assignment problem described in section 2. Computational results with an early (and somewhat inefficient) version of this code [10] show that for sparse assignment problems, AUCTION outperforms by a large margin the code by Jonker and Volgenant [20]. This latter code is a two-phase method; the first phase is an extensive initialization procedure based on the relaxation method of [4] and the second phase is a sequential shortest path method. Our own experiments indicate also that for

sparse problems, AUCTION outperforms by a large margin other types of two-phase methods which combine the relaxation method with a Hungarian-type or sequential shortest path method.

(2) RELAXII (written by Bertsekas and Tseng [12]): This is a state-of-the-art code for general linear minimum cost network flow problems, based on the relaxation method [5,11].

(3) RNET (written by Grigoriadis and Hsu): This is a state-of-the-art code for general linear minimum cost network flow problems, based on the simplex method.

In analogy with earlier auction algorithms, TRANSAUCTION applies the algorithm of the previous section with successively smaller values of ϵ, starting from a large value and ending with $\epsilon = 1/\min\{M, N\}$; the price vector obtained at the end of each application of the algorithm is used as the starting price vector for the next application of the algorithm. The idea of successive reduction of ϵ is known as *ϵ-scaling* and has been suggested in the original proposal of the auction algorithm as a method of improving performance. ϵ-scaling was analyzed first in [16] (and more fully in [17]), in the broader context of the ϵ-relaxation method of [6,7], where it was shown that it leads to polynomial algorithms. By introducing appropriate data structures and ϵ-scaling, and by combining the complexity analysis of the unscaled ϵ-relaxation method [6], and of scaling ([16,17] and also [9,10]), it is possible to use the algorithm of the previous section to construct an $O((M + N)^3 \log(C \min\{M, N\}))$ transportation algorithm, where $C = \max\{|a_{ij}| \mid j \in A(i)\}$. This is demonstrated in more general form in a report by the authors, which will be published shortly. Our TRANSAUCTION code has polynomial complexity, but does not use all the data structures needed to attain the polynomial complexity bound just mentioned; it is doubtful that an implementation attaining this bound would perform better than TRANSAUCTION in practice. The details of the ϵ-scaling scheme that we used are somewhat complicated. Roughly, all cost coefficients a_{ij} are first multiplied with $\min\{M, N\}$, so that the threshold value of ϵ that guarantees optimality is $\epsilon = 1$; then ϵ is initialized at a value of $C \min\{M, N\}/2$ for the first application of the algorithm of the previous section; ϵ is reduced by a certain factor (4–6 are recommended values) with each successive application of the algorithm until the final value $\epsilon = 1$ is reached. There is also an additional feature, called *adaptive scaling*, whereby the value of ϵ is gradually modified before the algorithm terminates based on some heuristic rules. Adaptive scaling is also used optionally in the public domain version of the AUCTION code.

The test problems we used were of two types. The first type are problems generated randomly with the public domain program NETGEN [22]. Figure 6 shows the times required by AUCTION and TRANSAUCTION for solving NETGEN assignment problems of different sizes. The figure reflects the additional overhead (between 2 and 3) which is required to maintain the data structures used by the implementation of the auction algorithm for transportation problems.

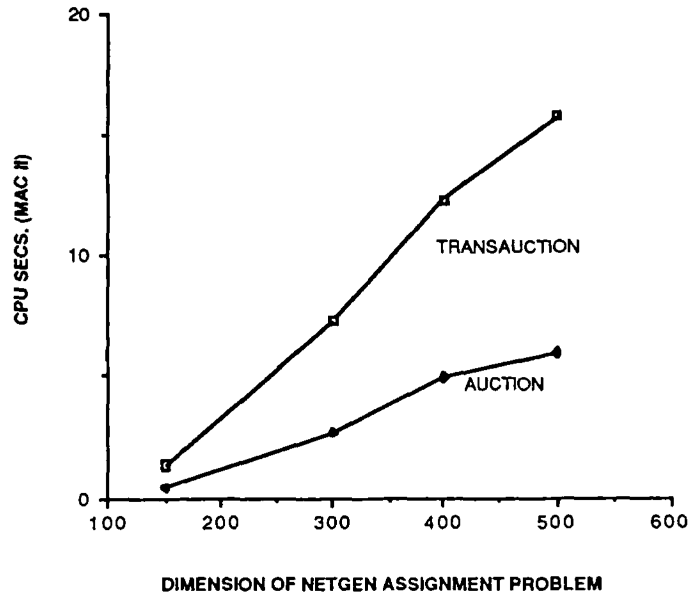

Fig. 6. Comparison of AUCTION and TRANSAUCTION for NETGEN assignment problems of different size. The number of arcs in each problem was 12.5% of maximum.

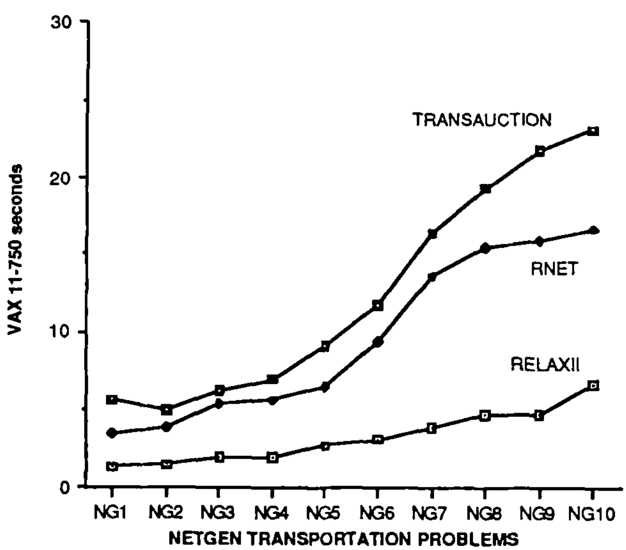

Fig. 7. Performance of TRANSAUCTION, RELAXII and RNET algorithms on NETGEN transportation problem benchmarks.

Figure 7 illustrates the computation times required by the TRANSAUCTION, RELAXII and RNET codes to solve the first ten standard problems of [22]. (These tests were done on a VAX 11-750 because we do not have a version of RNET that runs on an MAC II.) Here we have symmetric transportation problems, with the number of supply nodes ranging from 100 to 150 and the number of arcs ranging from 1300 to 6300. Figure 7 shows that TRANSAUCTION runs slower than RELAXII and roughly comparable with RNET. The total supply in these problems is 1000 times the number of sources.

The second type of problems that we tested are asymmetric transportation problems with relatively few levels of supplies and demands; by this we mean that the sources (sinks) can be divided into a few groups with roughly comparable values of supply (demand) within each group. Problems arise often in practice, where few sources with large supplies are allocated to many sinks with small demands. They are the type of problems for which TRANSAUCTION outperforms substantially both RELAXII and RNET. This is supported by the results shown in figs. 8–10. For these problems, the TRANSAUCTION code obtains an optimal solution in 20–50% of the time required by RELAXII. Figures 8–10 indicate that the advantage of TRANSAUCTION over the other codes increases with problem dimension. The problems of these figures are representative of the results obtained from many test runs with qualitatively similar problems.

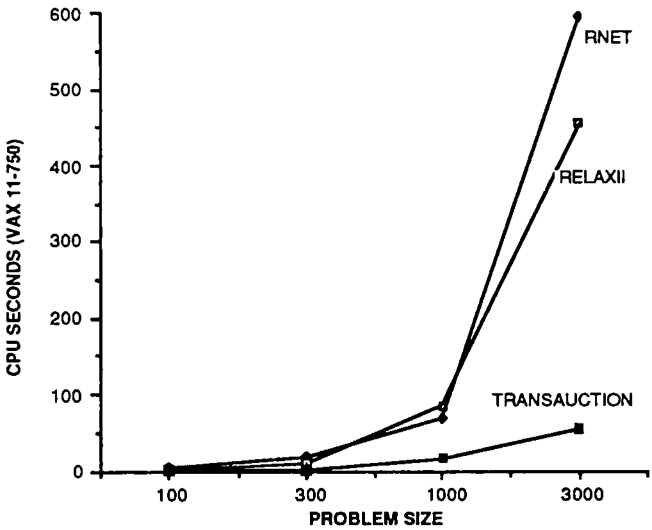

Fig. 8. Performance of TRANSAUCTION, RELAXII and RNET on transportation-assignment problems. All problems have 100 sinks, 10 with large demand and 90 with small demand. Problem size is described by the number of sources, all of which have unit supply. The number of arcs in each problem is 14% of maximum.

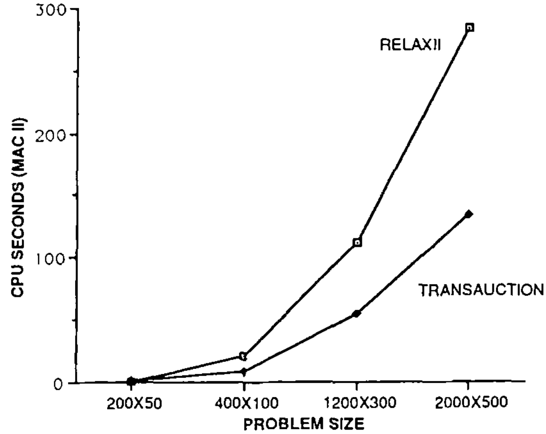

Fig. 9. Performance of TRANSAUCTION and RELAXII on transportation problems of homogeneous type. The number of sources equals four times the number of sinks. The supply of each source ranges from 1 – 9, with average 5. The sinks are divided into two classes, with 10% of the sinks having 50% of demand divided evenly, while 90% of the sinks divide the remaining 50% of demand evenly. The average number of arcs in each problem is 5% of maximum.

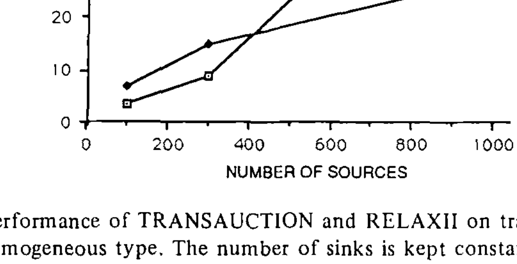

Fig. 10. Performance of TRANSAUCTION and RELAXII on transportation problems of homogeneous type. The number of sinks is kept constant at 100, with 10 sinks with demand 225 and 90 sinks with demand 25. The average number of arcs in each problem is 14% of maximum. The supplies of the sources were generated according to a uniform distribution over [1, 100] (for the 100-source problem), [1, 30] (for the 300-source problem), and [1, 10] (for the 1000-source problem), with corrections made to ensure that the total demand was equal to the total supply.

7. Conclusions

The auction algorithm of this paper is consistent with the current trend of solving linear programming problems using ideas from nonlinear programming. It has performed better than all transportation codes available to us on various classes of asymmetric transportation problems with relatively few levels of suuplies and demands. The algorithm is suitable for parallel implementation; parallel versions have already been developed, but this work lies outside the scope of this paper. The algorithm can also be extended to solve general linear minimum cost flow problems in $O(N^3 \log(NC))$ time. This extension, together with its complexity analysis, will be given in a future publication.

References

[1] M.L. Balinski, A competitive (dual) simplex method for the assignment problem, Math. Progr. 34(1986)125.

[2] R. Barr, F. Glover and D. Klingman, The alternating basis algorithm for assignment problems, Math. Progr. 13(1977)1.

[3] D.P. Bertsekas, A distributed algorithm for the assignment problem, Laboratory for Information and Decision Systems Unpublished Report, M.I.T. (March 1979).

[4] D.P. Bertsekas, A new algorithm for the assignment problem, Math. Progr. 21(1981)152.

[5] D.P. Bertsekas, A unified framework for primal-dual methods in minimum cost network flow problems, Math. Progr. 32(1985)125.

[6] D.P. Bertsekas, Distributed asynchronous relaxation methods for linear network flow problems, LIDS Report P-1601, M.I.T. (Sept. 1986; revised Nov. 1986).

[7] D.P. Bertsekas, Distributed relaxation methods for linear network flow problem, *Proc. 25th IEEE Conf. on Decision and Control,* Athens, Greece (1986) pp. 2101 – 2106.

[8] D.P. Bertsekas, The auction algorithm: A distributed relaxation method for the assignment problem, LIDS Report P-1653, M.I.T. (March 1987); Also in: Ann. Oper. Res. 14(1988)105.

[9] D.P. Bertsekas and J. Eckstein, Distributed asynchronous relaxation methods for linear network flow problems, *Proc. IFAC '87,* Munich, Germany (July 1987).

[10] D.P. Bertsekas and J. Eckstein, Dual coordinate step methods for linear network flow problems, Math. Progr. Series B 42(1988)203.

[11] D.P. Bertsekas and P. Tseng, Relaxation methods for minimum cost ordinary and generalized network flow problems, LIDS Report P-1462, M.I.T. (May 1985); Oper. Res. 36(1988)93.

[12] D.P. Bertsekas and P. Tseng, RELAX: A code for linear network flow problems, in: *FORTRAN Codes for Network Optimization,* ed. B. Simeone et al., Ann. Oper. Res. 13 (1988)125.

[13] D.P. Bertsekas and J.N. Tsitsiklis, *Parallel and Distributed Computation: Numerical Methods* (Prentice-Hall, Englewood Cliffs, NJ, 1989).

[14] M. Engquist, A successive shortest path algorithm for the assignment problem, INFOR 20 (1982)370.

[15] F. Glover, R. Glover and D. Klingman, Threshold assignment algorithm, Center for Business Decision Analysis Report CBDA 107, Graduate School of Business, University of Texas at Austin (Sept. 1982).

[16] A.V. Goldberg, Solving minimum-cost flow problems by successive approximations, extended abstract, submitted to STOC 87, Nov. 1986.

[17] A.V. Goldberg and R.E. Tarjan, Solving minimum cost flow problems by successive approximation, *Proc. 19th ACM STOC* (May, 1987).

[18] D. Goldfarb, Efficient dual simplex methods for the assignment problem, Math. Progr. 33(1985)187.

[19] M. Hung, A polynomial simplex method for the assignment problem, Oper. Res. 31(1983) 595.

[20] R. Jonker and A. Volgenant, A shortest augmenting path algorithm for dense and sparse linear assignment problems, Computing 28(1987)325.

[21] J. Kennington and R. Helgason, *Algorithms for Network Programming* (Wiley, New York, 1980).

[22] D. Klingman, A. Napier and J. Stutz, NETGEN – A program for generating large scale (un)capacitated assignment, trasnportation, and minimum cost flow network problems, Management Science 20(1974)814.

[23] H.W. Kuhn, The Hungarian method for the assignment problem, Naval Research Logistics Quarterly 2(1955)83.

[24] L.F. McGinnis, Implementation and testing of a primal-dual algorithm for the assignment problem, Oper. Res. 31(1983)277.

[25] C.H. Papadimitriou and K. Steiglitz, *Combinatorial Optimization: Algorithms and Complexity* (Prentice-Hall, Englewood Cliffs, NJ, 1982).

[26] R.T. Rockafellar, *Network Flows and Monotropic Programming* (Wiley, New York, 1984).