SIAM J. OPTIMIZATION
Vol. 3, No. 2, pp. 268-297, May 1993

004

REVERSE AUCTION AND THE SOLUTION OF INEQUALITY CONSTRAINED ASSIGNMENT PROBLEMS*

DIMITRI P. BERTSEKAS†, DAVID A. CASTAÑON‡, AND HARALAMPOS TSAKNAKIS§

Abstract. In this paper auction algorithms for solving several types of assignment problems with inequality constraints are proposed. Included are asymmetric problems with different numbers of persons and objects, and multiassignment problems, where persons may be assigned to several objects and vice versa. A central new idea in all these algorithms is to combine regular auction, where persons bid for objects by raising their prices, with reverse auction, where objects compete for persons by essentially offering discounts. Reverse auction can also be used to accelerate substantially (and sometimes dramatically) the convergence of regular auction for symmetric assignment problems.

Key words. assignment problem, network optimization, auction, linear programming

AMS subject classifications. primary, 90C47; secondary, 90C05

1. Introduction. Let us consider the classical symmetric assignment problem where we want to match n persons and n objects on a one-to-one basis. The benefit for matching a person with an object is given, and we want to assign all persons to distinct objects so as to maximize the total benefit. The auction algorithm is a method for solving this problem that was first proposed in [Ber79], and was subsequently developed in [Ber85], [Ber88], and [BeE88]. It operates like a real-life auction. There is a price for each object, and at each iteration, unassigned persons bid simultaneously for their "best" objects (the ones offering maximum benefit minus price), thereby raising the corresponding prices. Objects are then awarded to the highest bidder. The bidding increments must be at least equal to a positive parameter ε, and are chosen so as to preserve an ε-complementary slackness property. For good theoretical as well as practical performance, it may be important to use ε-scaling, which consists of applying the algorithm several times, starting with a large value of ε and successively reducing ε up to an ultimate value that is less than some threshold ($1/n$ when a_{ij} are integer). Each scaling phase provides good initial prices for the next. For tutorial presentations of the auction algorithm, we refer to [Ber90], [Ber91], and [Ber92a].

We note that there are several extensions of the auction algorithm, e.g., to transportation problems [BeC89a] and to minimum cost flow problems (the ε-relaxation method of [Ber86a] and [Ber86b], and the network auction algorithm of [BeC89b]). Computational studies on serial and parallel machines [BeC89b], [BeC89c], [CSW89], [Cas92], [KKZ89], [PhZ88], [WeZ90], [WeZ91], [Zak90] have shown that the algorithm is very effective, particularly for sparse symmetric assignment problems and special types of transportation problems.

In this paper we consider several new extensions of the auction algorithm for variations of the assignment problem described above. For some of these problems, no effective adaptation of the auction algorithm has been known so far, while for other

* Received by the editors March 23, 1991; accepted for publication (in revised form) March 23, 1992. This work was supported in part by the BM/C3 Technology branch of the United States Army Strategic Defense Command.

† Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, Massachusetts 02139.

‡ Department of Electrical Engineering, Boston University, and ALPHATECH, Inc., Burlington, Massachusetts 01803.

§ ALPHATECH, Inc., Burlington, Massachusetts 01803.

problems, including the symmetric assignment problem, the ideas of this paper have resulted in auction algorithms with substantially improved performance over the ones previously known.

Central to the present paper is an alternative form of the auction algorithm, called *reverse auction*, where, roughly, the *objects* compete for persons by *lowering* their prices. In particular, objects decrease their prices to a level that is sufficiently low to lure a person away from his/her currently held object. We can show that forward and reverse auctions are mathematically equivalent, but their combination results in algorithms that can solve problems that forward or reverse auction by themselves either cannot solve at all or can solve but much more slowly.

In the next section, we show how to combine forward and reverse auctions to solve symmetric assignment problems. In particular we provide mechanisms for switching gracefully between the two types of auction, using a special type of ε-complementary slackness condition. As shown by computational results given in § 5, the combined forward/reverse method substantially outperforms the regular (forward) method. The reason appears to be that the combined method suffers much less from "price wars," that is, protracted bid sequences involving a small number of persons competing for a smaller number of objects using small bidding increments. In fact, it may not be necessary to resort to ε-scaling, involving the solution of several subproblems, to improve the performance of the method.

In § 3, we consider asymmetric assignment problems, where the number of persons is less than the number of objects. As a result, in a feasible assignment, we require that every person, but not necessarily every object, be assigned. The original paper on the auction algorithm [Ber79] showed that this problem can be solved by the auction algorithm provided the prices of all objects start at zero. This approach is often very effective in practice, particularly when the number of persons is much less than the number of objects, but unfortunately it precludes the use of ε-scaling. As a result, it is ineffective for problems where price wars are likely to arise. By suitably combining forward and reverse auctions, we eliminate this drawback. In particular, we give a new auction algorithm for solving the asymmetric assignment problem, where the starting object prices can be arbitrary, so that ε-scaling can be used in the same way as for symmetric problems.

In § 4, we consider an interesting class of assignment-like problems, called *mutiassignment problems*, which arise in multitarget tracking applications (see the comments of § 4). There are no specialized network flow methods that can solve these problems at present, although they can be solved by general purpose network methods such as primal-simplex, primal-dual, or relaxation methods. We develop new classes of auction algorithms for multiassignment problems by combining the ideas of forward and reverse auctions.

Finally, in § 5, we present computational results using various experimental codes implementing the new algorithms of this paper. For each of the problems considered (symmetric and asymmetric assignment, and two types of multiassignment problems), we show that the new methods of this paper substantially (and often dramatically) outperform current state-of-the-art codes.

2. Reverse auction for symmetric assignment problems. In the symmetric assignment problem there are n persons and n objects. The benefit or value of assigning person i to object j is a_{ij}. The set of objects to which person i can be assigned is a nonempty set denoted $A(i)$. An *assignment* S is a (possibly empty) set of person-object pairs (i, j) such that $j \in A(i)$ for all $(i, j) \in S$; for each person i there can be at most one pair

$(i,j) \in S$; and for every object j there can be at most one pair $(i,j) \in S$. Given an assignment S, we say that person i is *assigned* if there exists a pair $(i,j) \in S$; otherwise we say that i is *unassigned.* We use similar terminology for objects. An assignment is said to be *feasible* if it contains n pairs, so that every person and every object is assigned; otherwise the assignment is called *partial.* We want to find an assignment $\{(1,j_1),\ldots,(n,j_n)\}$ with maximum total benefit $\sum_{i=1}^{n} a_{ij_i}$.

The auction algorithm for the symmetric assignment problem proceeds iteratively and terminates when a feasible assignment is obtained. At the start of the generic iteration we have a partial assignment S and a price vector $p=(p_1,\ldots,p_n)$ satisfying *ε-complementary slackness* (ε-CS). This is the condition

$$a_{ij}-p_j \geqq \max_{k\in A(i)}\{a_{ik}-p_k\}-\varepsilon \quad \forall (i,j)\in S. \tag{1}$$

As an initial choice, one can use an arbitrary set of prices together with the empty assignment, which trivially satisfies ε-CS. The iteration consists of two phases: the *bidding phase* and the *assignment phase,* described in the following.

BIDDING PHASE. Let I be a nonempty subset of persons i that are unassigned under the assignment S. For each person $i \in I$:

(1) Find a "best" object j_i having maximum value, that is,

$$j_i = \arg\max_{j\in A(i)}\{a_{ij}-p_j\},$$

and the corresponding value

$$v_i = \max_{j\in A(i)}\{a_{ij}-p_j\}, \tag{2}$$

and find the best value offered by objects other than j_i,

$$w_i = \max_{j\in A(i),\, j\neq j_i}\{a_{ij}-p_j\}. \tag{3}$$

(If j_i 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_i.)

(2) Compute the "bid" of person i given by

$$b_{ij_i} = p_{j_i} + v_i - w_i + \varepsilon = a_{ij_i} - w_i + \varepsilon. \tag{4}$$

(We characterize this situation by saying that person i bid for object j_i, and that object j_i received a bid from person i. The algorithm works if the bid has any value between $p_{j_i}+\varepsilon$ and $p_{j_i}+v_i-w_i+\varepsilon$, but it tends to work fastest for the maximal choice of (4).)

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,

$$p_j := \max_{i\in P(j)} b_{ij}, \tag{5}$$

remove from the assignment S any pair (i,j) (if j was assigned to some i under S), and add to S the pair (i_j,j), where i_j is a person in $P(j)$ attaining the maximum above.

Note that there is some freedom in choosing the subset of persons I that bid during an iteration. One possibility is to let I consist of a single unassigned person. This version, known as the *Gauss-Seidel version* in view of its similarity with Gauss-Seidel methods for solving systems of nonlinear equations, usually works best in a serial computing environment. The version where I consists of all unassigned persons

is the one best suited for parallel computation, and is known as the *Jacobi version*, in view of its similarity with Jacobi methods for solving systems of nonlinear equations.

The choice of bidding increment $v_i - w_i + \varepsilon$ for a person i [cf. (4)] is such that ε-CS is preserved, as stated in the following well-known proposition.

PROPOSITION 1. *The auction algorithm preserves* ε-CS *throughout its execution; that is, 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.*

Proof. See [Ber79], [Ber88], [BT89], or [Ber91] for the proof.

Furthermore, the algorithm is valid in the sense stated below.

PROPOSITION 2. *If at least one feasible assignment exists, the auction algorithm terminates in a finite number of iterations with a feasible assignment that is within* $n\varepsilon$ *of being optimal (and is optimal if the problem data is integer and* $\varepsilon < 1/n$).

Proof. See [Ber79], [Ber88], [BeT89], or [Ber91] for the proof.

The auction algorithm can be shown to have an $O(A(n + nC/\varepsilon))$ worst-case running time, where A is the number of arcs of the assignment graph, and

$$C = \max_{(i,j)\in\mathcal{A}} |a_{ij}|$$

is the maximum absolute object value; see [Ber79], [BeE88], and [BeT89]. Thus, the amount of work needed to solve the problem can depend strongly on the value of ε as well as of C. In practice, the dependence of the running time on ε and C is often significant, particularly for sparse problems.

To obtain polynomial complexity, we can use *ε-scaling*, which consists of applying the algorithm several times, starting with a large value of ε and successively reducing ε up to an ultimate value that is less than $1/n$. Each application of the algorithm, called a *scaling phase*, provides good initial prices for the next application. For integer data, it can be shown that the worst-case running time of the auction algorithm using scaling and appropriate data structures is $O(nA \log(nC))$; see [BeE88] and [BeT89]. We note that while ε-scaling was suggested in the original proposal of the auction algorithm [Ber79], it was first analyzed in [Gol87] (see also [GoT90]) in the context of the ε-relaxation method. This minimum cost flow algorithm (also known as preflow-push) was proposed in [Ber86a] and [Ber86b], and is essentially equivalent to the auction algorithm [Ber92b]. Not much is known about the average complexity of the auction algorithm. However, an interesting analysis of [Sch90] suggests that for uniformly distributed arc costs its running time grows proportionally to something like $A \log n$ or $A \log n \log(nC)$; this is roughly consistent with computational results using randomly generated problems.

2.1. Reverse auction. In the auction algorithm, persons compete for objects by bidding and raising the price of their best object. It is possible to use an alternative form of the auction algorithm, called *reverse auction*, where *objects* compete for persons. In particular, objects decrease their prices to a level that is sufficiently low to either attract an unassigned person or lure a person away from its currently held object.

In order to describe reverse auction, we introduce a *profit* variable π_i for each person i. The role that profits play for persons is analogous to the role prices play for objects. We can describe the reverse auction algorithm in two equivalent ways: one where unassigned objects lower their prices as much as possible to attract a person without violating ε-CS, and another where unassigned objects select a best person and raise his/her profit as much as possible without violating ε-CS. For analytical convenience, we will adopt the second description rather than the first.

Let us consider the following ε-CS condition for a (partial) assignment S and a profit vector π:

$$a_{ij} - \pi_i \geqq \max_{k \in B(j)} \{a_{kj} - \pi_k\} - \varepsilon \quad \forall (i,j) \in S, \tag{6}$$

where $B(j)$ is the set of persons that can be assigned to object j,

$$B(j) = \{i \mid (i,j) \in \mathcal{A}\}.$$

For feasibility, we assume that this set is nonempty for all j. Note the symmetry of this condition with the corresponding one for prices; cf. (1). The reverse auction algorithm starts with and maintains an assignment and a profit vector π satisfying the above ε-CS condition. It terminates when the assignment is feasible. At the beginning of each iteration, we have an assignment S and a profit vector π satisfying the ε-CS condition (6).

TYPICAL ITERATION OF REVERSE AUCTION. Let J be a nonempty subset of objects j that are unassigned under the assignment S. For each object $j \in J$:

(1) Find a "best" person i_j such that

$$i_j = \arg \max_{i \in B(j)} \{a_{ij} - \pi_i\},$$

and the corresponding value

$$\beta_j = \max_{i \in B(j)} \{a_{ij} - \pi_i\}, \tag{7}$$

and find

$$\omega_j = \max_{i \in B(j), i \neq i_j} \{a_{ij} - \pi_i\}. \tag{8}$$

(If i_j is the only person in $B(j)$, we define ω_j to be $-\infty$ or, for computational purposes, a number that is much smaller than β_j.)

(2) Each object $j \in J$ bids for person i_j an amount

$$b_{i_j j} = \pi_{i_j} + \beta_j - \omega_j + \varepsilon = a_{i_j j} - \omega_j + \varepsilon. \tag{9}$$

(3) For each person i that received at least one bid, increase π_i to the highest bid

$$\pi_i := \max_{j \in P(i)} b_{ij}, \tag{10}$$

where $P(i)$ is the set of objects from which i received a bid; remove from the assignment S any pair (i, j) (if i was assigned to some j under S), and add to S the pair (i, j_i), where j_i is an object in $P(i)$ attaining the maximum above.

Note that reverse auction is identical to (forward) auction with the roles of persons and objects, as well as profits and prices, interchanged. Thus, by using the corresponding (forward) auction result (cf. Proposition 2), we have the following.

PROPOSITION 3. *If at least one feasible assignment exists, the reverse auction algorithm terminates in a finite number of iterations. The feasible assignment obtained upon termination is within $n\varepsilon$ of being optimal (and is optimal if the problem data are integer and $\varepsilon < 1/n$).*

2.2. Combined forward and reverse auction. One of the reasons we are interested in reverse auction is to construct algorithms that switch from forward to reverse auction and back. Such algorithms must simultaneously maintain a price vector p satisfying the ε-CS condition (1) and a profit vector π satisfying the ε-CS condition (6). To this

end we introduce an ε-CS condition for the *pair* (π, p), which, as we will see, implies the other two. Maintaining this condition is essential for switching gracefully between forward and reverse auction.

DEFINITION 1. An assignment S and a pair (π, p) are said to satisfy ε-CS if

$$\pi_i + p_j \geqq a_{ij} - \varepsilon \quad \forall (i,j) \in \mathcal{A}, \tag{11a}$$

$$\pi_i + p_j = a_{ij} \quad \forall (i,j) \in S. \tag{11b}$$

We have the following proposition.

PROPOSITION 4. *Suppose that an assignment S, together with a profit-price pair (π, p) satisfies ε-CS. Then*

(a) *S and π satisfy the ε-CS condition*

$$a_{ij} - \pi_i \geqq \max_{k \in B(j)} \{a_{kj} - \pi_k\} - \varepsilon \quad \forall (i,j) \in S. \tag{12}$$

(b) *S and p satisfy the ε-CS condition*

$$a_{ij} - p_j \geqq \max_{k \in A(i)} \{a_{ik} - p_k\} - \varepsilon \quad \forall (i,j) \in S. \tag{13}$$

(c) *If S is feasible, then S is within $n\varepsilon$ of being an optimal assignment.*

Proof. (a) In view of (11b), for all $(i,j) \in S$, we have $p_j = a_{ij} - \pi_i$, so (11a) implies that $a_{ij} - \pi_i \geqq a_{kj} - \pi_k - \varepsilon$ for all $k \in B(j)$. This shows (12).

(b) The proof is the same as that of part (a) with the roles of π and p interchanged.

(c) Since by part (b), the ε-CS condition (13) is satisfied, by Proposition 2, S is within $n\varepsilon$ of being optimal. □

We now introduce a combined forward/reverse algorithm. The algorithm starts with and maintains an assignment S and a profit-price pair (π, p) satisfying the ε-CS condition (11). It terminates when the assignment is feasible.

COMBINED FORWARD/REVERSE AUCTION ALGORITHM.

Step 1 (run forward auction): Execute several iterations of the forward auction algorithm (subject to the termination condition), and at the end of each iteration (after increasing the prices of the objects that received a bid), set

$$\pi_i = a_{ij_i} - p_{j_i}, \tag{14}$$

for every person-object pair (i, j_i) that entered the assignment during the iteration. Go to Step 2.

Step 2 (run reverse auction): Execute several iterations of the reverse auction algorithm (subject to the termination condition), and at the end of each iteration (after increasing the profits of the persons that received a bid), set

$$p_j = a_{i_j j} - \pi_{i_j}, \tag{15}$$

for every person-object pair (i_j, j) that entered the assignment during the iteration. Go to Step 1.

Note that the additional overhead of the combined algorithm over the forward or the reverse algorithm is minimal; just one update of the form (14) or (15) is required per iteration for each object or person that received a bid during the iteration. An alternative but probably less efficient possibility is to update the profits π_i of the assigned persons via (14) (or the prices p_j of the assigned objects via (15)) just before switching to reverse auction (or forward auction, respectively). An important property is that the updates of (14) and (15) maintain the ε-CS condition (11) for the pair

(π, p), and therefore, by Proposition 4, maintain the required ε-CS conditions (12) and (13) for π and p, respectively. This is shown in the following proposition.

PROPOSITION 5. *If the assignment and profit-price pair available at the start of an iteration of either the forward or the reverse auction algorithm satisfy the* ε-CS *condition* (11), *the same is true for the assignment and profit-price pair obtained at the end of the iteration, provided* (14) *is used to update* π (*in the case of forward auction*), *and* (15) *is used to update* p (*in the case of reverse auction*).

Proof. Assume for concreteness that forward auction is used, and let (π, p) and $(\bar{\pi}, \bar{p})$ be the profit-price pair before and after the iteration, respectively. Then, $\bar{p}_j \geqq p_j$ for all j (with strict inequality if and only if j received a bid during the iteration). Therefore, we have $\bar{\pi}_i + \bar{p}_j \geqq a_{ij} - \varepsilon$ for all (i, j) such that $\pi_i = \bar{\pi}_i$. Furthermore, we have $\bar{\pi}_i + \bar{p}_j = \pi_i + p_j = a_{ij}$ for all (i, j) that belong to the assignment before as well as after the iteration. Also, in view of the update (14), we have $\bar{\pi}_i + \bar{p}_{j_i} = a_{ij_i}$ for all pairs (i, j_i) that entered the assignment during the iteration. What remains is to verify that the condition

$$\bar{\pi}_i + \bar{p}_j \geqq a_{ij} - \varepsilon \quad \forall j \in A(i) \tag{16}$$

holds for all persons i that submitted a bid and were assigned to an object, say j_i, during the iteration. Indeed, for such a person i, we have by (4),

$$\bar{p}_{j_i} = a_{ij_i} - \max_{j \in A(i), j \neq j_i} \{a_{ij} - p_j\} + \varepsilon,$$

which implies that

$$\bar{\pi}_i = a_{ij_i} - \bar{p}_{j_i} \geqq a_{ij} - p_j - \varepsilon \geqq a_{ij} - \bar{p}_j - \varepsilon \quad \forall j \in A(i).$$

This shows the desired relation (16). □

Note that during forward auction, the object prices p_j increase, while the profits π_i decrease, but exactly the opposite happens in reverse auction. For this reason, the termination proof used for forward auction (see, e.g., [BeT89, p. 371]) does not apply to the combined method. Indeed, it is possible to construct examples of feasible problems where the combined method never terminates if the switch between forward and reverse auctions is done arbitrarily. However, it is easy to guarantee that the combined algorithm terminates finitely for a feasible problem; it is sufficient to ensure that some "irreversible progress" is made before switching between forward and reverse auction. One easily implementable possibility is to refrain from switching until at least one more person-object pair has been added to the assignment. In this way there can be a switch at most $(n-1)$ times between the forward and reverse steps of the algorithm. Since for a feasible problem, forward and reverse auction by themselves have guaranteed finite termination, the final step will terminate with a feasible assignment satisfying ε-CS.

The combined forward/reverse auction algorithm often works substantially faster than the forward version. It seems to be affected less by "price wars," that is, protracted sequences of small price rises by a number of persons bidding for a smaller number of objects. Price wars can still occur in the combined algorithm, but they arise through more complex and unlikely problem structures than in the forward algorithm. For this reason the combined forward/reverse auction algorithm depends less on ε-scaling for good performance than its forward counterpart. One consequence of this is that starting with $\varepsilon = 1/n$ and bypassing ε-scaling is often the best choice. Another consequence is that a larger ε-reduction factor can typically be used with no price war effects in ε-scaled forward/reverse auction than in ε-scaled forward auction. As a result, fewer

ε-scaling phases are typically needed in forward/reverse auction to deal effectively with price wars.

3. Auction algorithms for asymmetric assignment problems. Reverse auction can be used in conjunction with forward auction to provide algorithms for solving the asymmetric assignment problem, where the number of objects n is larger than the number of persons m. Here we still require that each person be assigned to some object, but we allow objects to remain unassigned. As before, an assignment S is a (possibly empty) set of person-object pairs (i,j) such that $j \in A(i)$ for all $(i,j) \in S$; for each person i there can be at most one pair $(i,j) \in S$; and for every object j there can be at most one pair $(i,j) \in S$. The assignment S is said to be feasible if all persons are assigned under S.

The corresponding linear programming problem is

$$\begin{aligned} &\text{maximize} && \sum_{(i,j)\in\mathcal{A}} a_{ij}x_{ij} \\ &\text{subject to} \\ &&& \sum_{j\in A(i)} x_{ij} = 1 \quad \forall i = 1,\dots,m, \\ &&& \sum_{i\in B(j)} x_{ij} \leqq 1 \quad \forall j = 1,\dots,n, \\ &&& 0 \leqq x_{ij} \quad \forall (i,j) \in \mathcal{A}. \end{aligned} \tag{17}$$

We can convert this program to the minimum cost flow problem

$$\begin{aligned} &\text{minimize} && \sum_{(i,j)\in\mathcal{A}} (-a_{ij})x_{ij} \\ &\text{subject to} \\ &&& \sum_{j\in A(i)} x_{ij} = 1 \quad \forall i = 1,\dots,m, \\ &&& \sum_{i\in B(j)} x_{ij} + x_{sj} = 1 \quad \forall j = 1,\dots,n, \\ &&& \sum_{j=1}^{n} x_{sj} = n - m, \\ &&& 0 \leqq x_{ij} \quad \forall (i,j) \in \mathcal{A}, \\ &&& 0 \leqq x_{sj} \quad \forall j = 1,\dots,n, \end{aligned} \tag{18}$$

by replacing maximization by minimization, by reversing the sign of a_{ij}, and by introducing a supersource node s, which is connected to each object node j by an arc (s,j) of zero cost and feasible flow range $[0,\infty)$.

Using the duality theory for minimum cost network flow problems (see, e.g., [BeT89, p. 335] or [Ber91, p. 35]), it can be verified that the corresponding dual problem is

$$\begin{aligned} &\text{minimize} && \sum_{i=1}^{m} \pi_i + \sum_{j=1}^{n} p_j - (n-m)\lambda \\ &\text{subject to} && \pi_i + p_j \geqq a_{ij} \quad \forall (i,j) \in \mathcal{A}, \\ &&& \lambda \leqq p_j \quad \forall j = 1,\dots,n, \end{aligned} \tag{19}$$

where we have converted maximization to minimization, we have used $-\pi_i$ in place of the price of each person node i, and we have denoted by λ the price of the supersource node s.

We now introduce an ε-CS condition for an assignment S and a pair (π, p).

DEFINITION 2. An assignment S and a pair (π, p) are said to satisfy ε-CS if

$$\pi_i + p_j \geqq a_{ij} - \varepsilon \quad \forall (i,j) \in \mathcal{A}, \tag{20a}$$

$$\pi_i + p_j = a_{ij} \quad \forall (i,j) \in S, \tag{20b}$$

$$p_j \leqq \min_{k:\text{ assigned under } S} p_k \quad \forall j \quad \text{unassigned under } S. \tag{20c}$$

The following proposition clarifies the significance of the preceding ε-CS condition.

PROPOSITION 6. *If a feasible assignment S satisfies the ε-CS conditions* (20) *together with a pair (π, p), then S is within $m\varepsilon$ of being optimal for the asymmetric assignment problem. The triplet $(\hat{\pi}, \hat{p}, \lambda)$, where*

$$\lambda = \min_{k:\text{ assigned under } S} p_k, \tag{21a}$$

$$\hat{\pi}_i = \pi_i + \varepsilon \quad \forall i = 1, \ldots, m, \tag{21b}$$

$$\hat{p}_j = \begin{cases} p_j & \text{if } j \text{ is assigned under } S, \\ \lambda & \text{if } j \text{ is unassigned under } S \end{cases} \quad \forall j = 1, \ldots, n, \tag{21c}$$

is within $m\varepsilon$ of being an optimal solution of the dual problem (19).

Proof. For any feasible assignment $\{(i, k_i) | i = 1, \ldots, m\}$ and for any triplet $(\bar{\pi}, \bar{p}, \lambda)$ satisfying the dual feasibility constraints $\bar{\pi}_i + \bar{p}_j \geqq a_{ij}$ for all $(i,j) \in \mathcal{A}$ and $\lambda \leqq \bar{p}_j$ for all j, we have

$$\sum_{i=1}^{m} a_{ik_i} \leqq \sum_{i=1}^{m} \bar{\pi}_i + \sum_{i=1}^{m} \bar{p}_{k_i} \leqq \sum_{i=1}^{m} \bar{\pi}_i + \sum_{j=1}^{n} \bar{p}_j - (n-m)\lambda.$$

By maximizing over all feasible assignments $\{(i, k_i) | i = 1, \ldots, m\}$ and by minimizing over all dual-feasible triplets $(\bar{\pi}, \bar{p}, \lambda)$, we see that

$$A* \leqq D*,$$

where $A*$ is the optimal assignment value and $D*$ is the minimal dual cost.

Let now $S = \{(i, j_i) | i = 1, \ldots, m\}$ be the given assignment satisfying ε-CS together with (π, p), and consider the triplet $(\hat{\pi}, \hat{p}, \lambda)$ defined by (21). Since for all i, we have $\hat{\pi}_i + \hat{p}_{j_i} = a_{ij} + \varepsilon$, we obtain

$$A* \geqq \sum_{i=1}^{m} a_{ij_i} = \sum_{i=1}^{m} \hat{\pi}_i + \sum_{i=1}^{m} \hat{p}_{j_i} - m\varepsilon = \sum_{i=1}^{m} \hat{\pi}_i + \sum_{j=1}^{n} \hat{p}_j - (n-m)\lambda - m\varepsilon \geqq D* - m\varepsilon,$$

where the last inequality holds because the triplet $(\hat{\pi}, \hat{p}, \lambda)$ is feasible for the dual problem. Since we showed earlier that $A* \leqq D*$, the desired conclusion follows. □

Consider now trying to solve the asymmetric assignment problem by means of auction. We can start with any assignment S and pair (π, p) satisfying the first two ε-CS conditions (20a) and (20b), and perform a forward auction (as defined earlier for the symmetric assignment problem) up to the point where each person is assigned to a distinct object. For a feasible problem, it can be seen that this will yield, in a finite number of iterations, a feasible assignment S satisfying the first two conditions (20a) and (20b). If we select initially all object prices to be zero, then upon termination of the algorithm, the prices of the unassigned objects will still be at zero, while the

prices of the assigned objects will be nonnegative. Therefore, the ε-CS condition (20c) will also be satisfied, and by Proposition 6 the assignment S obtained will be optimal. Unfortunately, the use of zero initial prices precludes the use of ε-scaling, and leaves the method susceptible to price wars. To be able to use ε-scaling we must be able to use arbitrary initial prices, but then the assignment S obtained by forward auction may not be optimal because the prices of the unassigned objects may not be minimal, that is, they may not satisfy the third ε-CS condition (20c). Roughly, what is happening here is that forward auction cannot resolve whether the objects that were left unassigned upon termination are intrinsically "undesirable" because they offer relatively low benefit to the persons, or whether they were left unassigned because their initial prices were high relative to the initial prices of the assigned objects.

To resolve this dilemma, we use a modified form of reverse auction to lower the prices of the objects that were left unassigned upon termination of the forward auction. After several reverse auction iterations in which persons may be reassigned to other objects, the third condition (20c) will be satisfied. We will show that the assignment thus obtained satisfies all the ε-CS conditions (20a)-(20c) and by Proposition 6 is optimal within $m\varepsilon$ (and thus optimal if the problem data are integer and $\varepsilon < 1/m$).

The modified reverse auction starts with a feasible assignment S and with a pair (π, p) satisfying the first two ε-CS conditions (20a) and (20b). (For a feasible problem, such an S and (π, p) can be obtained by regular forward or reverse auction, as discussed earlier.) Let us denote by λ the minimal assigned object price under the initial assignment,

$$\lambda = \min_{j:\ \text{assigned under the initial assignment } S} p_j. \tag{22}$$

The typical iteration of modified reverse auction is the same as the one of reverse auction, except that only unassigned objects j with $p_j > \lambda$ participate in the auction. In particular, the algorithm maintains a feasible assignment S and a pair (π, p) satisfying (20a) and (20b), and terminates when all unassigned objects j satisfy $p_j \leqq \lambda$, in which case it will be seen that the third ε-CS condition (20c) will be satisfied as well. The scalar λ will be kept fixed throughout the algorithm.

TYPICAL ITERATION OF MODIFIED REVERSE AUCTION FOR ASYMMETRIC ASSIGNMENT. Select an object j that is unassigned under the assignment S, and satisfies $p_j > \lambda$ (if no such object can be found, the algorithm terminates). Find a "best" person i_j such that

$$i_j = \arg \max_{i \in B(j)} \{a_{ij} - \pi_i\},$$

and the corresponding value

$$\beta_j = \max_{i \in B(j)} \{a_{ij} - \pi_i\}, \tag{23}$$

and find

$$\omega_j = \max_{i \in B(j), i \neq i_j} \{a_{ij} - \pi_i\}. \tag{24}$$

(If i_j is the only person in $B(j)$, we define ω_j to be $-\infty$.) If $\lambda \geqq \beta_j - \varepsilon$, set $p_j := \lambda$ and go to the next iteration. Otherwise, let

$$\delta = \min \{\beta_j - \lambda, \beta_j - \omega_j + \varepsilon\}. \tag{25}$$

Set

$$p_j := \beta_j - \delta, \tag{26}$$

$$\pi_{i_j} := \pi_{i_j} + \delta, \tag{27}$$

add to the assignment S the pair (i_j, j), and remove from S the pair (i_j, j'), where j' is the object that was assigned to i_j under S at the start of the iteration.

Note that the formula (25) for the bidding increment δ is such that the object j enters the assignment at a price which is no less than λ (and is equal to λ if and only if the minimum in (25) is attained by the first term). Furthermore, we have $\delta \geqq \varepsilon$ (when δ is calculated, that is, when $\lambda > \beta_j - \varepsilon$), so it can be seen from (26) and (27) that throughout the algorithm, prices are monotonically decreasing and profits are monotonically increasing. The following proposition establishes the validity of the method.

PROPOSITION 7. *The modified reverse auction algorithm for the asymmetric assignment problem terminates in a finite number of iterations and the assignment obtained is within $m\varepsilon$ of being optimal.*

Proof. In view of Proposition 6, the result will follow once we prove the following:

(a) The modified reverse auction iteration preserves the first two ε-CS conditions (20a) and (20b), as well as the condition

$$\lambda \leqq \min_{j:\ \text{assigned under the current assignment } S} p_j, \tag{28}$$

so upon termination of the algorithm (necessarily with the prices of all unassigned objects less or equal to λ), the third ε-CS condition (20c) is satisfied.

(b) The algorithm terminates finitely.

We will prove these facts in sequence.

We assume that the conditions (20a), (20b), and (28) are satisfied at the start of an iteration, and we will show that they are also satisfied at the end of the iteration. First, consider the case where there is no change in the assignment, which happens when $\lambda \geqq \beta_j - \varepsilon$. Then (20b) and (28) are automatically satisfied at the end of the iteration; only p_j changes in the iteration according to

$$p_j := \lambda \geqq \beta_j - \varepsilon = \max_{i \in B(j)} \{a_{ij} - \pi_i\} - \varepsilon,$$

so the condition (20a) is also satisfied at the end of the iteration.

Next consider the case where there is a change in the assignment during the iteration. Let (π, p) and $(\bar{\pi}, \bar{p})$ be the profit-price pair before and after the iteration, respectively, and let j and i_j be the object and person involved in the iteration. By construction [cf. (26) and (27)], we have $\bar{\pi}_{i_j} + \bar{p}_j = a_{i_j j}$, and since $\bar{\pi}_i = \pi_i$ and $\bar{p}_k = p_k$ for all $i \neq i_j$ and $k \neq j$, we see that the condition (20b) ($\bar{\pi}_i + \bar{p}_k = a_{ik}$) is satified for all assigned pairs (i, k) at the end of the iteration.

To show that the condition (20a) is satisfied at the end of the iteration, that is,

$$\bar{\pi}_i + \bar{p}_k \geqq a_{ik} - \varepsilon \quad \forall (i, k) \in \mathcal{A}, \tag{29}$$

consider first objects $k \neq j$. Then, $\bar{p}_k = p_k$ and since $\bar{\pi}_i \geqq \pi_i$ for all i, the above condition holds, since at the start of the iteration, we have $\pi_i + p_k \geqq a_{ik} - \varepsilon$ for all (i, k). Consider next the case $k = j$. Then, condition (29) holds for $i = i_j$, since $\bar{\pi}_{i_j} + \bar{p}_j = a_{i_j j}$. Also using (23)-(26) and the fact $\delta \geqq \varepsilon$, we have for all $i \neq i_j$,

$$\begin{aligned} \bar{\pi}_i + \bar{p}_j = \pi_i + \bar{p}_j &\geqq \pi_i + \beta_j - (\beta_j - \omega_j + \varepsilon) \\ &= \pi_i + \omega_j - \varepsilon \geqq \pi_i + (a_{ij} - \pi_i) - \varepsilon = a_{ij} - \varepsilon, \end{aligned}$$

so condition (29) holds for $i \neq i_j$ and $k = j$, completing the proof.

To see that condition (28) is maintained by the iteration, note that by (23), (24), and (26), we have

$$\bar{p}_j = \beta_j - \delta \geqq \beta_j - (\beta_j - \lambda) = \lambda.$$

Finally, to show that the algorithm terminates finitely, we note that in the typical iteration involving object j and person i_j, there are two possibilities:

(1) The price of object j is set to λ without the object entering the assignment; this occurs if $\lambda \geqq \beta_j - \varepsilon$.

(2) The profit of person i_j increases by at least ε (this is seen from the definition (25) of δ; we have $\lambda < \beta_j - \varepsilon$ and $\beta_j \geqq \omega_j$, so $\delta \geqq \varepsilon$).

Since only objects j with $p_j > \lambda$ can participate in the auction, possibility (1) can occur only a finite number of times. Thus, if the algorithm does not terminate, the profits of some persons will increase to ∞. This is impossible, since when person i is assigned to object j, we must have by (20b) and (28)

$$\pi_i = a_{ij} - p_j \leqq a_{ij} - \lambda,$$

so the profits are bounded from above by $\max_{(i,j)\in\mathcal{A}} a_{ij} - \lambda$. Thus the algorithm must terminate finitely. □

As mentioned earlier, forward auction followed by modified reverse auction can start with arbitrary initial prices. As a result, one can use ε-scaling, performing a sequence of auctions with decreasing values of ε. This can be shown to improve the theoretical worst-case complexity of the method, and is often beneficial in practice, particularly for sparse problems. Out of several possible variations of the method, the one we have tested most uses the modified reverse auction only in the last ε-scaling phase. In all other ε-scaling phases just forward auction is used.

Reverse auction also can be used to solve the variation of the two-sided inequality constrained assignment problem, where persons (as well as objects) need not be assigned if this degrades the assignment's value. This problem can be converted to an asymmetric assignment problem where all persons must be assigned by introducing for each person i an artificial object i' and a zero cost arc (i, i'). One can then use the algorithm given earlier to solve this problem. The algorithm can be streamlined so that the calculations involving the artificial objects and arcs are handled efficiently.

4. Auction algorithms for multiassignment problems. An interesting type of assignment problem is described by the linear program

$$\begin{aligned} &\text{maximize} \quad \sum_{(i,j)\in\mathcal{A}} a_{ij}x_{ij} \\ &\text{subject to} \\ &\sum_{j\in\mathcal{A}(i)} x_{ij} \geqq 1 \quad \forall i = 1, \ldots, m, \\ &\sum_{i\in B(j)} x_{ij} = 1 \quad \forall j = 1, \ldots, n, \\ &0 \leqq x_{ij} \quad \forall (i,j) \in \mathcal{A}, \end{aligned} \tag{30}$$

where $m < n$. For feasibility, we assume that the sets $A(i)$ and $B(j)$ are nonempty for all i and all j, respectively. This is known as the *multiassignment* problem, and is characterized by the possibility of assignment of more than one object to a single person; such a person is said to be *multiassigned.* Problems of this type arise in military applications, such as multitarget tracking with sensors of limited resolution [Bla86], where objects correspond to tracked vehicles and persons correspond to data points, each representing at least one vehicle (but possibly more than one, because of the sensor's limited resolution). The multiassignment problem results when we try to associate data points with vehicles so as to match as closely as possible these data points with our prior knowledge of the vehicles' positions.

We can convert the multiassignment problem to the minimum cost flow problem

$$\begin{aligned}
&\text{minimize} \quad \sum_{(i,j)\in\mathcal{A}} (-a_{ij})x_{ij} \\
&\text{subject to} \\
&\sum_{j\in A(i)} x_{ij} - x_{si} = 1 \quad \forall i = 1, \ldots, m, \\
&\sum_{i\in B(j)} x_{ij} = 1 \quad \forall j = 1, \ldots, n, \qquad (31) \\
&\sum_{i=1}^{m} x_{si} = n - m, \\
&0 \leqq x_{ij} \quad \forall (i,j) \in \mathcal{A}, \\
&0 \leqq x_{si} \quad \forall i = 1, \ldots, n,
\end{aligned}$$

by replacing maximization by minimization, by reversing the sign of a_{ij}, and by introducing a supersource node s, which is connected to each person node i by an arc (s, i) of zero cost and feasible flow range $[0, \infty)$ (see Fig. 1).

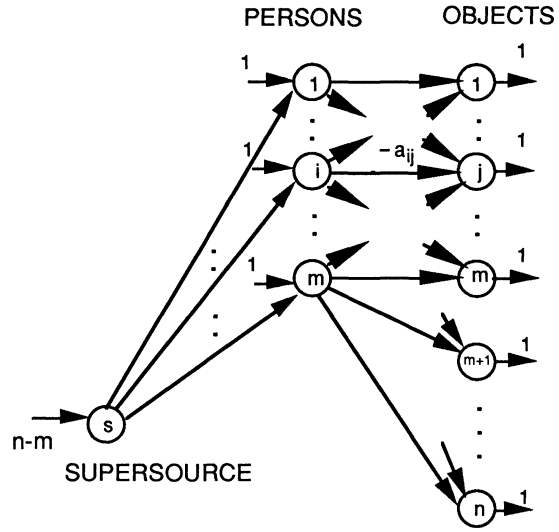

FIG. 1. *Converting a multiassignment problem into a minimum cost flow problem involving a supersource node s and a zero cost artificial arc* (s, i) *with feasible flow range* $[0, \infty)$ *for each person i.*

Using duality theory again and appropriately redefining the price variables corresponding to the nodes, it can be verified that the corresponding dual problem is

$$\begin{aligned}
&\text{minimize} \quad \sum_{i=1}^{m} \pi_i + \sum_{j=1}^{n} p_j + (n-m)\lambda \\
&\text{subject to} \quad \pi_i + p_j \geqq a_{ij} \quad \forall (i,j) \in \mathcal{A}, \qquad (32) \\
&\qquad\qquad \lambda \geqq \pi_i \quad \forall i = 1, \ldots, m.
\end{aligned}$$

We now introduce an ε-CS condition for an assignment S and a pair (π, p).

DEFINITION 3. A multiassignment S and a pair (π, p) are said to satisfy ε-CS if

$$\pi_i + p_j \geqq a_{ij} - \varepsilon \quad \forall (i,j) \in \mathcal{A}, \tag{33a}$$

$$\pi_i + p_j = a_{ij} \quad \forall (i,j) \in S, \tag{33b}$$

$$\pi_i = \max_{k=1,\ldots,m} \pi_k \quad \text{if } i \text{ is multiassigned under } S. \tag{33c}$$

We have the following result.

PROPOSITION 8. *Assume that the benefits a_{ij} are integer. If a feasible assignment S satisfies the ε-CS conditions* (33) *together with a pair (π, p) for $\varepsilon < 1/m$, then S is optimal for the multiassignment problem.*

Proof. If S is not optimal, there must exist a cycle Y in the equivalent network of Fig. 1 with no repeated nodes along which the assignment S can be modified to result in a new feasible assignment S' with improved primal cost. Assume for the moment that the supersource s is in the cycle; thus, let Y be

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

In the above cycle, the nodes i_q represent distinct persons, the nodes j_q represent distinct objects and

$$(i_q, j_q) \in S, \quad j_q \in A(i_{q-1}), \quad (i_{q-1}, j_q) \notin S, \quad q = 2, \ldots, k.$$

Augmentation along Y results in replacing the pairs $(i_q, j_q) \in S$, $q = 2, \ldots, k$, by the pairs (i_{q-1}, j_q), $q = 2, \ldots, k$, in the assignment. It can be seen that i_k must be multiassigned prior to the augmentation; the reason is that with the augmentation along Y, the arc (i_k, j_k) will exit the assignment, so person i_k will be left unassigned and feasibility will be violated after the augmentation. Because Y has no repeated nodes, we have $k \leqq m$, which, based on the hypothesis, implies $k\varepsilon < 1$.

Since the augmentation results in strict cost improvement and the benefits are integer, we must have

$$\sum_{q=2}^{k} a_{i_q j_q} + 1 \leqq \sum_{q=2}^{k} a_{i_{q-1} j_q},$$

or equivalently,

$$\sum_{q=2}^{k} (a_{i_q j_q} - p_{j_q}) + 1 \leqq \sum_{q=2}^{k} (a_{i_{q-1} j_q} - p_{j_q}).$$

Using the above relation and the ε-CS condition (33a), it follows that

$$\sum_{q=2}^{k} \pi_{i_q} + 1 = \sum_{q=2}^{k} (a_{i_q j_q} - p_{j_q}) + 1 \leqq \sum_{q=2}^{k} (a_{i_{q-1} j_q} - p_{j_q}) \leqq \sum_{q=1}^{k-1} \pi_{i_q} + (k-1)\varepsilon.$$

From this relation, we obtain

$$1 - (k-1)\varepsilon \leqq \pi_{i_1} - \pi_{i_k}.$$

This is a contradiction because we argued earlier that $k\varepsilon < 1$, and that i_k is multiassigned, which implies that $\pi_{i_k} \geqq \pi_{i_1}$ (cf. (33c)).

If Y does not contain s, a similar argument establishes the result. □

Consider now trying to solve the multiassignment problem by means of auction. We can start with any assignment S and profit-price pair (π, p) satisfying the first two ε-CS conditions (33a) and (33b), and perform a forward auction up to the point where each person is assigned to a (single) distinct object, while satisfying the conditions (33a) and (33b). However, this assignment will not be feasible, because some objects will still be unassigned.

To make further progress, we use a modified reverse auction, which starts with the final results of the forward auction, that is, with an assignment S, where each person is assigned to a single distinct object, and with a pair (π, p) satisfying the first two ε-CS conditions (33a) and (33b). Let us denote by λ the maximal initial person profit,

$$\lambda = \max_{i=1,\ldots,m} \pi_i. \tag{34}$$

The typical iteration, given below, is the same as the one of reverse auction, except that unassigned objects j that bid for a person may not necessarily displace the object assigned to the person but may instead *share* the person with its already assigned object(s). In particular, the algorithm maintains an assignment S, for which each person is assigned to at least one object, and a pair (π, p) satisfying (33a) and (33b); it terminates when all unassigned objects j have been assigned. It will be seen that upon termination, the third ε-CS condition (33c) will be satisfied as well. The scalar λ is kept fixed throughout the algorithm.

TYPICAL ITERATION OF MODIFIED REVERSE AUCTION FOR MULTIASSIGNMENT. Select an object j that is unassigned under the assignment S (if all objects are assigned, the algorithm terminates). Find a "best" person i_j such that

$$i_j = \arg \max_{i \in B(j)} \{a_{ij} - \pi_i\}, \tag{35}$$

and the corresponding value

$$\beta_j = \max_{i \in B(j)} \{a_{ij} - \pi_i\}, \tag{36}$$

and find

$$\omega_j = \max_{i \in B(j), i \neq i_j} \{a_{ij} - \pi_i\}. \tag{37}$$

(If i_j is the only person in $B(j)$, we define ω_j to be $-\infty$.) Let

$$\delta = \min \{\lambda - \pi_{i_j}, \beta_j - \omega_j + \varepsilon\}. \tag{38}$$

Add (i_j, j) to the assignment S, set

$$p_j := \beta_j - \delta, \tag{39}$$

$$\pi_{i_j} := \pi_{i_j} + \delta \tag{40}$$

and if $\delta > 0$, remove from the assignment S the pair (i_j, j'), where j' was assigned to i_j under S.

Note that in an iteration, the number of assigned objects increases by one if and only if $\delta = 0$ (which is equivalent to $\pi_{i_j} = \lambda$, since the second term $\beta_j - \omega_j + \varepsilon$ in (38) is always greater than or equal to ε). The following proposition establishes the validity of the method.

PROPOSITION 9. *The modified reverse auction algorithm for the multiassignment problem with integer benefits terminates in a finite number of iterations with an optimal assignment when* $\varepsilon < 1/m$.

Proof In view of Proposition 8, the result will follow once we prove the following:

(a) The modified reverse auction iteration preserves the ε-CS conditions (33), as

well as the condition

$$\lambda = \max_{i=1,\ldots,m} \pi_i. \tag{41}$$

(b) The algorithm terminates finitely (necessarily with a feasible assignment).

To show (a) above, we use induction. In particular, we show that if the conditions (33) and (41) are satisfied at the start of an iteration, they are also satisfied at the end of the iteration. Indeed, this is easily seen to be true for (33a) and (33b). Equations (33c) and (41) are preserved since we have $\lambda = \max_{i=1,\ldots,m} \pi_i$ at the start of the iteration and the only profit that changes is π_{i_j}, which by (38) and (40) is set to something that is less than or equal to λ, and is set to λ if and only if i_j is multiassigned at the end of the iteration.

To show finite termination, we observe that a person i can receive a bid only a finite number of times after the profit π_i is set to λ, since at each of these times the corresponding object will get assigned to i without any object already assigned to i becoming unassigned. On the other hand, by (38) and (40), at an iteration where a person i receives a bid, the profit π_i is either set equal to λ or else increases by at least ε. Since profits are bounded above by λ throughout the algorithm, it follows that each person can receive only a finite number of bids, proving finite termination. □

4.1. Two-sided multiassignment problem. There are several variations of the multiassignment problem and the preceding algorithm. For example, the problem where there is an upper bound α_i on the number of objects person i can be assigned to, that is,

$$\begin{aligned} &\text{maximize} \quad \sum_{(i,j)\in\mathcal{A}} a_{ij}x_{ij} \\ &\text{subject to} \\ &1 \leqq \sum_{j\in A(i)} x_{ij} \leqq \alpha_i \quad \forall i = 1,\ldots,m, \\ &\sum_{i\in B(j)} x_{ij} = 1 \quad \forall j = 1,\ldots,n, \\ &0 \leqq x_{ij} \quad \forall (i,j)\in\mathcal{A}, \end{aligned} \tag{42}$$

where α_i are given integers. This multiassignment problem admits solution by a similar auction algorithm as the preceding one; we will not give the details.

Another interesting variation of the multiassignment problem arises when objects, as well as persons, can be multiassigned, up to a certain limit. This problem, referred to as *two-sided multiassignment*, can be written as

$$\begin{aligned} &\text{maximize} \quad \sum_{(i,j)\in\mathcal{A}} a_{ij}x_{ij} \\ &\text{subject to} \\ &\sum_{j\in A(i)} x_{ij} \geqq 1 \quad \forall i = 1,\ldots,m, \\ &1 \leqq \sum_{i\in B(j)} x_{ij} \leqq \alpha_j \quad \forall j = 1,\ldots,n, \\ &0 \leqq x_{ij} \leqq 1 \quad \forall (i,j)\in\mathcal{A}, \end{aligned} \tag{43}$$

where α_j are given integers less. Note that if $\alpha_j = 1$, this problem is identical to the earlier problem (30).

Again, the above problem can be converted to a minimum cost network flow problem

$$
\begin{aligned}
&\text{minimize} \quad \sum_{(i,j)\in\mathcal{A}} (-a_{ij}x_{ij}) \\
&\text{subject to} \\
&\sum_{j\in A(i)} x_{ij} - x_{si} = 1 \quad \forall i = 1, \ldots, m, \\
&\sum_{i\in B(j)} x_{ij} - x_{js} = 1 \quad \forall j = 1, \ldots, n, \\
&\sum_{i=1}^{m} x_{si} - \sum_{j=1}^{n} x_{js} = n - m, \\
&0 \leqq x_{ij} \leqq 1 \quad \forall (i,j) \in \mathcal{A}, \\
&0 \leqq x_{si} \quad \forall i = 1, \ldots, m, \\
&0 \leqq x_{js} \leqq \alpha_j - 1 \quad \forall j = 1, \ldots, n,
\end{aligned} \tag{44}
$$

by replacing maximization by minimization, by reversing the sign of a_{ij}, by introducing a supersource node s with supply $n-m$, an arc (s, i) for each person i of zero cost and feasible flow range $[0, \infty)$, and an arc (j, s) for each object node j of zero cost and feasible flow range $[0, \alpha_j - 1]$ (see Fig. 2).

Using duality theory and appropriately redefining the price variables corresponding to the nodes, it can be seen that the corresponding dual problem is

$$
\begin{aligned}
&\text{minimize} \quad \sum_{i=1}^{m} \pi_i + \sum_{j=1}^{n} (p_j + \max\{0, (p_j + \lambda)(\alpha_j - 1)\}) \\
&\qquad + \sum_{(i,j)\in\mathcal{A}} \max\{0, a_{ij} - p_j - \pi_i\} + (n-m)\lambda \\
&\text{subject to} \quad \lambda \geqq \pi_i \quad \forall i = 1, \ldots, m,
\end{aligned} \tag{45}
$$

where λ is the dual price of the supersource node s. The above dual problem is similar to the earlier dual problem (32), with the exception of the cost terms introduced by the upper bounds on the arcs.

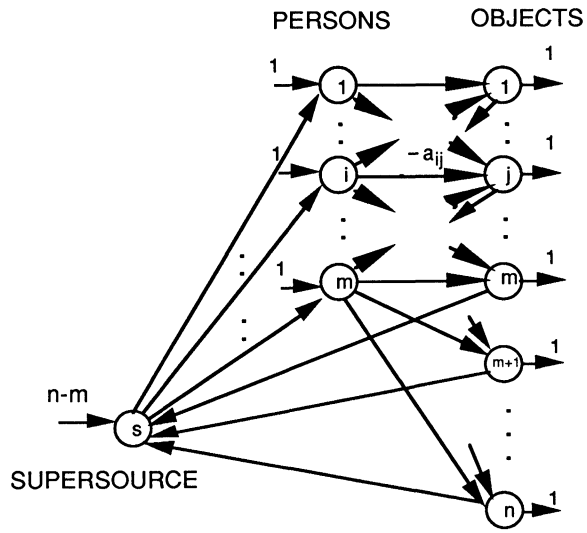

FIG. 2. *Converting a two-sided multiassignment problem into a minimum cost flow problem involving a supersource node s, zero cost artificial arcs* (s, i) *with feasible flow range* $[0, \infty)$ *for each person i, and zero cost artificial arcs* (j, s) *with feasible flow range* $[0, \alpha_j - 1]$ *for each object j.*

For the two-sided multiassignment problem, we introduce the following ε-CS condition for an assignment S and a pair (π, p).

DEFINITION 4. A multiassignment S and a pair (π, p) are said to satisfy ε-CS for the two-sided multiassignment problem if

$$\pi_i + p_j \geqq a_{ij} - \varepsilon \quad \forall (i,j) \in \mathcal{A}, \tag{46a}$$

$$\pi_i + p_j \leqq a_i \quad \forall (i,j) \in S, \tag{46b}$$

$$\pi_i = \max_{k=1,\ldots,m} \pi_k = \lambda \quad \text{if } i \text{ is multiassigned under } S, \tag{46c}$$

$$p_j + \lambda \geqq 0 \quad \text{if } j \text{ is multiassigned under } S, \tag{46d}$$

$$\text{if } p_j + \lambda > 0 \quad j \text{ must be assigned to } \alpha_j \text{ persons under } S. \tag{46e}$$

Using an argument similar to the proof of Proposition 8, we can establish the following result.

PROPOSITION 10. *Assume that the benefits a_{ij} are integers. If a feasible assignment S satisfies the ε-CS conditions* (46) *together wih a pair (π, p) for $\varepsilon < 1/m$, then S is optimal for the multiassignment problem.*

Proof. If S is not optimal, there must exist a cycle Y in the equivalent network of Fig. 2 with no repeated nodes along which the assignment S can be modified to result in a new feasible assignment S' with improved primal cost. There are five possible cases: (1) the cycle Y does not include node s; (2) the cycle Y includes s followed by a person and preceded by another person; (3) the cycle Y includes s followed by a person and preceded by an object; (4) the cycle Y inciudes s followed by an object and preceded by a person; and (5) the cycle Y includes s followed by an object and preceded by another object.

Assume for the moment that the node s is in the cycle and that it is followed and preceded by persons (case (2)); thus, let Y be

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

In order for augmentation along Y to result in a feasible assignment, we must have $(i_q, j_q) \in S$, $q = 2, \ldots, k$, $j_{q+1} \in A(i_q)$, $q = 1, \ldots, k-1$; furthermore, i_k must be multiassigned. Because Y has no repeated nodes, we have $k \leqq m$, which, based on the hypothesis, implies $k\varepsilon < 1$.

Augmentation along Y results in replacing the pairs (i_q, j_q), $q = 2, \ldots, k$, by the pairs (i_{q-1}, j_q), $q = 2, \ldots, k$, in the assignment. Since following augmentation along Y, the primal cost is strictly improved, we must have

$$\sum_{q=2}^{k} a_{i_q j_q} + 1 \leqq \sum_{q=2}^{k} a_{i_{q-1} j_q},$$

or equivalently,

$$\sum_{q=2}^{k} (a_{i_q j_q} - p_{j_q}) + 1 \leqq \sum_{q=2}^{k} (a_{i_{q-1} j_q} - p_{j_q}).$$

Using this relation, and the ε-CS conditions (46a) and (46b), we obtain

$$\sum_{q=1}^{k} \pi_{i_q} - \pi_{i_1} + 1 \leqq \sum_{q=2}^{k} (a_{i_q j_q} - p_{j_q}) + 1 \leqq \sum_{q=2}^{k} (a_{i_{q-1} j_q} - p_{j_q})$$

$$\leqq \sum_{q=1}^{k} \pi_{i_q} - \pi_{i_k} + (k-1)\varepsilon.$$

This yields

$$1 - (k-1)\varepsilon \leqq \pi_{i_1} - \pi_{i_k},$$

which is a contradiction because $k\varepsilon < 1$, and $\pi_{i_1} \leqq \pi_{i_k}$, since i_k is multiassigned (cf. (46c)).

Similarly, assume that node s is preceded and followed by an object in Y (case (5)); thus,

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

In order for augmentation along Y to produce a feasible assignment, we must have $(i_q, j_q) \in S$, $q = 1, \ldots, k$, $j_{q+1} \in A(i_q)$, $q = 1, \ldots, k-1$; furthermore, we must have $(i_{q-1}, j_q) \notin S$, $q = 2, \ldots, k+1$, j_1 must be multiassigned, and j_{k+1} must be assigned to less than $\alpha_{j_{k+1}}$ persons. Because Y has no repeated nodes, we have $k \leqq m$, which, based on the hypothesis, implies $k\varepsilon < 1$.

Augmentation along Y results in replacing the pairs (i_q, j_q), $q = 1, \ldots, k$, by the pairs (i_q, j_{q+1}), $q = 1, \ldots, k$, in the assignment; note that since j_1 is multiassigned and j_{k+1} can be assigned to at least one more person, the resulting modified assignment is feasible. Thus, we must have

$$\sum_{q=1}^{k} a_{i_q j_q} + 1 \leqq \sum_{q=1}^{k} a_{i_q j_{q+1}},$$

or equivalently,

$$\sum_{q=1}^{k} (a_{i_q j_q} - \pi_{i_q}) + 1 \leqq \sum_{q=1}^{k} (a_{i_q j_{q+1}} - \pi_{i_q}).$$

Using the above relation and the ε-CS conditions (46a) and (46b), we obtain

$$\sum_{q=1}^{k} p_{j_q} + 1 \leqq \sum_{q=1}^{k} (a_{i_q j_q} - \pi_{i_q}) + 1 \leqq \sum_{q=1}^{k} (a_{i_q j_{q+1}} - \pi_{i_q}) \leqq \sum_{q=1}^{k} p_{j_{q+1}} + k\varepsilon.$$

This yields $1 - k\varepsilon \leqq p_{j_{k+1}} - p_{j_1}$, which is a contradiction because $k\varepsilon < 1$, while by the CS conditions (46d) and (46e), we have $p_{i_{k+1}} = -\lambda \leqq p_{j_1}$ since j_{k+1} is assigned to less than α_{k+1} persons.

The proof for cases (1), (3), and (4) is similar. □

Consider now trying to solve the two-sided multiassignment problem using an auction algorithm. We start from any assignment S that has at most one person assigned to each object and at most one object assigned to each person, and a profit-price pair (π, p) satisfying the first two ε-CS conditions associated with regular auction (cf. (20a) and (20b)). We then use a forward auction algorithm up to the point where each person is assigned to a single (distinct) object, while satisfying the first two ε-CS conditions (46a) and (46b) (condition (46b) will actually be satisfied with equality). Note that this assignment will not be feasible since $m < n$.

At this point, we switch to using a modified reverse auction; denote by λ the maximal initial person profit

$$\lambda = \max_{i=1,\ldots,m} \pi_i. \tag{47}$$

Using this value of λ, we can determine which objects have prices p_j indicating that they can be multiassigned; in particular, the ε-CS condition (46e) suggests that any object with price p_j greater than $-\lambda$ should be assigned to as many persons as possible. In order to determine these persons, we use a reverse auction where each unassigned object, and each assigned object with price p_j greater than $-\lambda$ and assigned to less than α_j persons will bid to be assigned to an additional person. This reverse auction is modified in order to satisfy the ε-CS conditions at termination, as follows.

TYPICAL ITERATION OF MODIFIED REVERSE AUCTION FOR TWO-SIDED MULTI-ASSIGNMENT. Select an object j that is unassigned, or is assigned to at least one and less than α_j persons, and has p_j greater than $-\lambda$ (if no such object can be found, the algorithm terminates). If the set $\{i \in B(j) \mid (i,j) \notin S\}$ is empty, set $p_j = -\lambda$ and go to the next iteration. Otherwise, find a "best" person i_j such that

$$i_j = \arg \max_{i \in B(j),(i,j) \notin S} \{a_{ij} - \pi_i\}, \tag{48}$$

and the corresponding value

$$\beta_j = \max_{i \in B(j),(i,j) \notin S} \{a_{ij} - \pi_i\}, \tag{49}$$

and find

$$\omega_j = \max_{i \in B(j), i \neq i_j, (i,j) \notin S} \{a_{ij} - \pi_i\}. \tag{50}$$

(If the set $i \in B(j)$, $i \neq i_j$, $(i,j) \notin S$ is empty, we define ω_j to be $-\infty$.)

If j is unassigned, let

$$\delta = \min \{\lambda - \pi_{i_j}, \beta_j - \omega_j + \varepsilon\}. \tag{51a}$$

Add (i_j, j) to the assignment S, set

$$p_j := \omega_j - \varepsilon, \tag{51b}$$

$$\pi_{i_j} := \pi_{i_j} + \delta, \tag{51c}$$

and if $\delta > 0$, remove from the assignment S the pair (i_j, j'), where j' was assigned to i_j under S.

If j is assigned to at least one and less than α_j persons, and $p_j + \lambda > 0$, let

$$\delta = \min \{\lambda - \pi_{i_j}, \beta_j - \omega_j + \varepsilon, \beta_j + \lambda\}, \tag{52a}$$

and distinguish two cases:

(a) $\delta < \beta_j + \lambda$: In this case, add (i_j, j) to the assignment S, set

$$p_j := \max \{\omega_j - \varepsilon, -\lambda\}, \tag{52b}$$

$$\pi_{i_j} := \pi_{i_j} + \delta, \tag{52c}$$

$$\pi_i := \min \{a_{ij} - \max \{\omega_j - \varepsilon, -\lambda\}, \lambda\} \quad \forall i \text{ such that } (i,j) \in S, \tag{52d}$$

and if $\delta > 0$, remove from S the pair (i_j, j'), where j' was assigned to i_j under S.

(b) $\delta = \beta_j + \lambda$: In this case, set

$$p_j := -\lambda, \tag{53a}$$

$$\pi_{i_j} := \pi_{i_j} + \max \{0, \delta\}, \tag{53b}$$

$$\pi_i := \min \{a_{ij} + \lambda, \lambda\} \quad \forall i \text{ such that } (i,j) \in S, \tag{53c}$$

and, if $\delta > 0$, add (i_j, j) to the assignment S and remove from S the pair (i_j, j'), where j' was assigned to i_j under S.

Note that the above algorithm uses two types of iterations. The first type occurs when the bidding object is unassigned; then the number of unassigned objects decreases by one when δ is zero, which is equivalent to $\pi_i = \lambda$, so that person i can be multi-assigned. The second type of iteration occurs when the bidding object is already

assigned, but has price $p_j > -\lambda$; then either j is assigned to an additional person, or else p_j is reduced to the threshold price $-\lambda$.

The following proposition establishes the validity of the method.

PROPOSITION 11. *The modified reverse auction algorithm for the two-sided multi-assignment problem with integer benefits terminates in a finite number of iterations with an optimal assignment when* $\varepsilon < 1/m$.

Proof. In view of Proposition 10, the result will follow once we prove the following:

(a) The modified reverse auction iteration preserves the ε-CS conditions (46a)-(46d).

(b) The algorithm terminates finitely (necessarily with a feasible assignment).

(c) Upon termination, the ε-CS condition (46e) must be satisfied.

To show (a) above, we use induction. Let (π, p) and $(\bar{\pi}, \bar{p})$ be the profit-price pair before and after an iteration of the modified reverse auction algorithm, respectively, and let j and i_j be the object and person involved in the iteration. At the beginning of the first iteration, S and (π, p) satisfy

$$\pi_i + p_j \geqq a_{ij} - \varepsilon \quad \forall (i,j) \in \mathcal{A},$$

$$\pi_i + p_j = a_{ij} \quad \forall (i,j) \in S.$$

By construction, we also have

$$\pi_i \leqq \lambda \quad \forall i = 1, \ldots, m;$$

furthermore, every person is assigned to exactly one object, and every object is assigned to at most one person. Thus, the ε-CS conditions (46a)-(46d) are satisfied.

Assume that the ε-CS conditions (46a)-(46d) are satisfied at the beginning of an iteration. We consider three cases: (a) object j is currently unassigned, (b) object j is assigned and the bid increment δ satisfies $\delta < \beta_j + \lambda$, and (c) object j is assigned and the bid increment δ satisfies $\delta = \beta_j + \lambda$.

In case (a), by construction we have

$$\bar{\pi}_{i_j} + \bar{p}_j \leqq \pi_{i_j} + \beta_j - \omega_j + \varepsilon + p_j \leqq a_{i_j j}.$$

Furthermore, since π_{i_j} does not decrease, the ε-CS condition (46a) will be satisfied for all $k \in A(i_j)$, $k \neq j$. In addition, for any $i' \neq i_j$, $i' \in B(j)$, we have by (50)

$$\pi_{i'} + p_j = \pi_{i'} + \omega_j - \varepsilon \geqq a_{i'j} - \varepsilon,$$

establishing that the ε-CS conditions (46a) and (46b) are satisfied at the end of the iteration. In addition, the ε-CS condition (46c) is guaranteed to be satisfied by (51a) and (51c), and the ε-CS condition (46d) continues to be satisfied, since the prices of multiassigned objects were not affected.

In case (b), the price p_j, and the profits π_{i_j} and π_i, $(i,j) \in S$ are modified. Conditions (46c) and (46d) will be satisfied by the modified profits and prices at the end of the iteration by construction (cf. (52a)-(52d)). Assume $\delta = \beta_j - \omega_j + \varepsilon$; then $-\lambda \leqq \omega_j - \varepsilon$, so

$$\bar{p}_j = \max\{\omega_j - \varepsilon, -\lambda\} = \omega_j - \varepsilon \leqq p_j,$$

$$\bar{\pi}_{i_j} + \bar{p}_j = a_{ij} - \delta - \beta_j + \omega_j - \varepsilon = a_{ij},$$

establishing that the ε-CS condition (46b) holds for the new pair (i_j, j) entering the assignment. Similarly, the ε-CS condition (46b) is satisfied for all $(i,j) \in S$ by construction (cf. (52d)). Since $\bar{p}_j \leqq p_j$, equation (52d) implies that $\pi_i \leqq \bar{\pi}_i$ for all i, which in turn implies that the ε-CS condition (46a) is satisfied for all $k \in A(i)$, with $(i,k) \notin S$

and $(i,j) \in S$. Also, the ε-CS condition (46a) is satisfied for all $(i,j) \notin S$, $i \neq i_j$ because (50) implies

$$p_j = \omega_j - \varepsilon \geqq a_{ij} - \pi_i - \varepsilon \quad \forall (i,j) \notin S, i \neq i_j.$$

If, on the other hand, $\delta = \lambda - \pi_i$, then $a_{i_j j} \geqq 0$ and $\omega_j - \varepsilon \leqq a_{i_j j} - \lambda$. Thus, (46b) is satisfied for (i_j, j) at the end of the iteration, since

$$\bar{\pi}_{i_j} + \bar{p}_j = \lambda + \max\{-\lambda, \omega_j - \varepsilon\} \leqq a_{i_j j}.$$

Similarly, the ε-CS condition (46b) is satisfied for all $(i,j) \in S$ by (52d). Since $p_j + \lambda > 0$, p_j is not increased during the iteration. Thus, (52d) implies that $\pi_i \leqq \bar{\pi}_i$ for all i, so that the ε-CS condition (46a) is satisfied for all $k \in A(i)$, with $(i,k) \notin S$, and $(i,j) \in S$. Furthermore, since $p_j \geqq \omega_j - \varepsilon$, the ε-CS condition (46a) is satisfied for all $k \in B(j)$, with $(k,j) \notin S$.

In case (c), assume $\delta = \beta_j + \lambda > 0$. Then, the ε-CS condition (46b) is satisfied for the pair (i_j, j) because

$$\pi_{i_j j} + p_j = a_{i_j j} - \beta_j + \max\{0, \delta\} - \lambda = a_{i_j j}.$$

By assumption, the iteration decreases the price p_j and increases the profit π_{i_j}. Furthermore, (53c) implies that the profits $\pi_i \leqq \bar{\pi}_i$ for all i, so that the ε-CS condition (46a) is satisfied for all $(i,k) \in \mathcal{A}$, $(i,k) \notin S$, $(i,j) \in S$. Equation (53c) also guarantees that the ε-CS condition (46b) will be satisfied at the end of the iteration. If $\delta \leqq 0$, the assignment S is not modified; only the price p_j is decreased and the profits π_i, $(i,j) \in S$ are modified. The ε-CS condition (46b) is satisfied because of (53c); in addition, the ε-CS condition (46a) is satisfied because

$$0 \geqq \beta_j + \lambda \geqq a_{ij} - \pi_i + \lambda = a_{ij} - \pi_i - p_j \quad \forall (i,j) \notin S,$$

and the profits π_i, $i \in B(j)$, are nondecreasing.

The above arguments establish that the ε-CS conditions (46a)–(46d) are preserved by each modified reverse auction iteration. To complete the proof, we must show that the algorithm terminates finitely, and that at termination, the ε-CS condition (46e) is satisfied. It is easy to verify that the number of assigned pairs is nondecreasing, and, as shown above, the profits π_i are nondecreasing, while the prices p_j are nonincreasing. Furthermore, each iteration is guaranteed to produce one (or more) of the following three outcomes: (a) at least one profit π_i increases, (b) one additional pair is assigned, and (c) S remains unchanged, but the price p_j is set to the minimum value $-\lambda$. By construction, the profits π_i cannot rise above λ; furthermore, the prices p_j can only be reduced to $-\lambda$ once per object, and there is a finite maximum number of assigned pairs, which establishes finite termination. To show that the ε-CS condition (46e) is satisfied at termination, note that iterations occur until this condition is satisfied. This completes the proof. □

5. Numerical results. In this section we present some numerical results on the computational performance of the new auction algorithms described in the previous sections. The algorithms have been implemented in FORTRAN and have been compared with state-of-the-art algorithms for every class of problems considered in this paper.

5.1. Symmetric assignment problems. We first tested the two versions of forward/reverse auction (a scaled and an unscaled version) applied to symmetric assignment problems versus two other state-of-the-art codes: a forward auction code and the code of Jonker and Volgenant [JoV87]. The latter, abbreviated as JV code, consists

of two phases: an initialization phase, which is based on the naive auction algorithm (the forward auction algorithm with $\varepsilon = 0$), and a sequential shortest path method phase, which assigns the persons that are left unassigned by the initialization phase. It is widely believed that through the combination of the auction and the sequential shortest path algorithms, the JV code is substantially faster than the best pure sequential shortest path and Hungarian assignment codes (for some comparative evidence, see [Ber90]).

Our results for symmetric assignment problems are summarized in Figs. 3–6, where each data point represents an average over ten to thirty random problems with identical characteristics. In Figs. 3–6, a different characteristic (number of nodes, average node degree, and benefit range) of the problem was allowed to vary: the number of nodes in Fig. 3, the average node degree in Fig. 4, and the benefit range in Figs. 5 and 6. Experiments with problems of constant density and varying numbers of nodes and arcs have produced results that are qualitatively intermediate between the results of Figs. 3 and 4. Figures 5 and 6 are similar but they correspond to sparse and fully dense problems, respectively. It can be seen that the unscaled forward/reverse auction is running considerably faster than the other codes. The auction algorithms (remarkably, including the unscaled forward/reverse algorithm) are also quite insensitive to the benefit range; a similar conclusion regarding scaled forward auction was reached in [WeZ91]. Furthermore, all the auction codes run much faster than the JV code except when the problem is quite dense (cf. Fig. 4 when the number of arcs is large). Still, even for fully dense problems the unscaled forward/reverse algorithm is faster than the JV code, except when the benefit range is relatively small ([0, 100] in Fig. 6). There is an explanation for the excellent performance of the JV code for a fully dense problem with a small benefit range. What happens here is that the problem is solved essentially in the naive auction initialization phase of the code and the sequential shortest path phase plays no role. Thus, in this case, the JV code behaves like a very efficient auction algorithm.

In the test problems of Figs. 3–6 the arc benefits are uniformly distributed over the benefit range. In Fig. 7 we tested the effect of a two-level arc benefit distribution

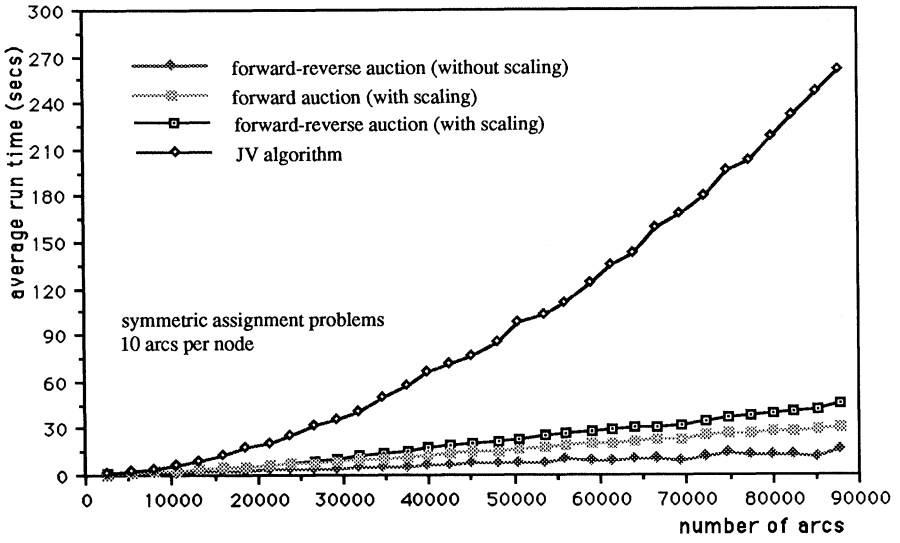

FIG. 3. *Run times for symmetric assignment problems on a* MAC II. *The degree of each person node is* 10. *Each data point represents an average of ten randomly generated problems. The arc benefits are drawn from the range* [0, 1000] *according to a uniform distribution.*

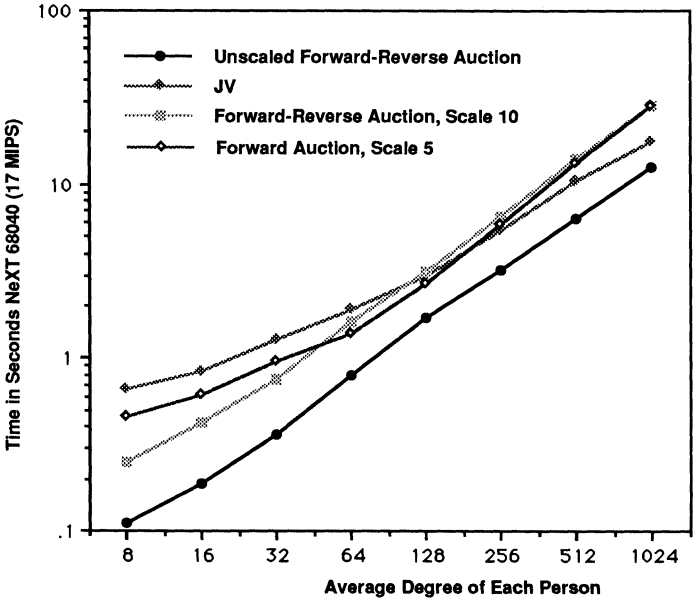

FIG. 4. *Run times for symmetric assignment problems on a* NeXT 68040. *The number of person nodes is* 1024 *and the average node degree varies. Each data point represents an average of* 30 *randomly generated problems. The arc benefits are drawn from the range* [0, 100000] *according to a uniform distribution.*

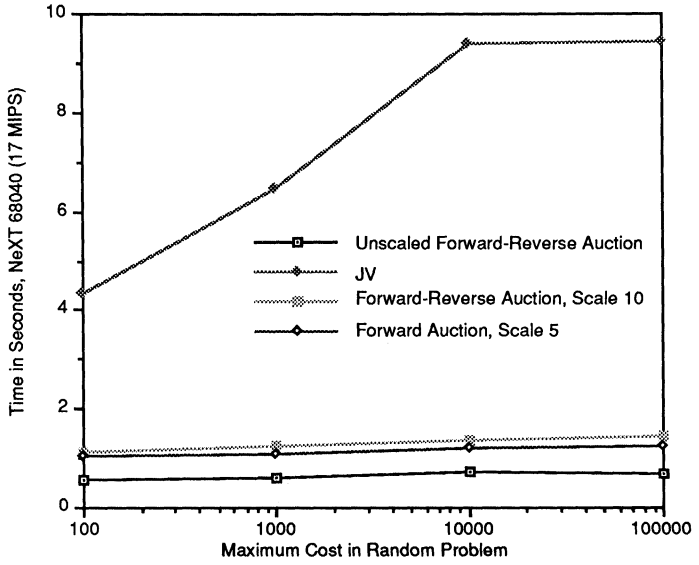

FIG. 5. *Run times for symmetric assignment problems on a* NeXT 68040. *The number of person nodes is* 4000 *and the degree of each is* 8. *Each data point represents an average of* 30 *randomly generated problems. The arc benefits are drawn from the range indicated according to a uniform distribution.*

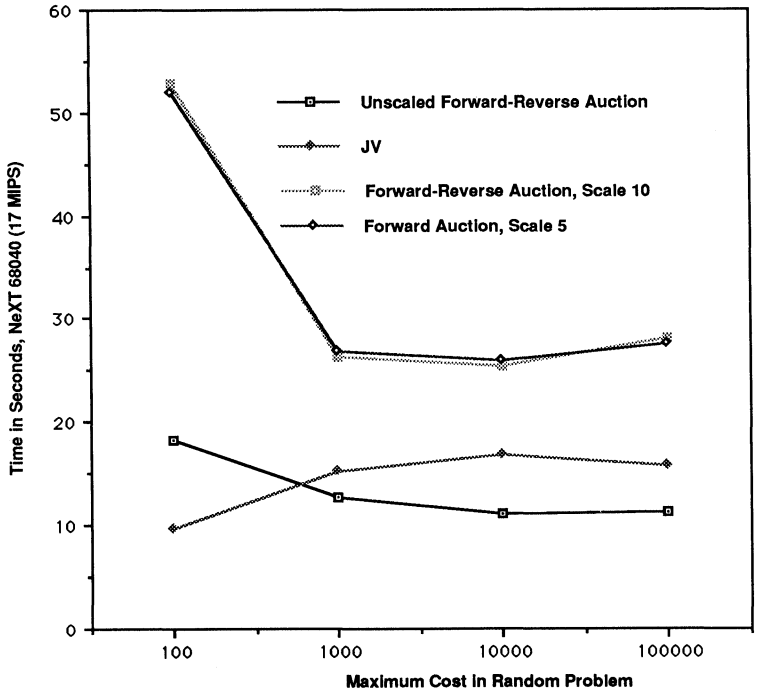

FIG. 6. *Run times for fully dense symmetric assignment problems with* 1024 *persons on a* NeXT 68040. *Each data point represents an average of* 30 *randomly generated problems. The arc benefits are drawn from the range indicated according to a uniform distribution.*

	FR Mean	FR St. Dev.	SFR10 Mean	SFR10 Std.	SF3 Mean	SF3 Std.	SF5 Mean	SF5 Std.	SF10 Mean	SF10 Std.
Easy	0.27	0.14	0.46	0.08	0.51	0.04	0.45	0.04	0.46	0.12
Difficult	0.25	0.05	1.15	0.09	1.77	0.11	1.91	0.32	2.99	0.40

FIG. 7. *Mean and standard deviation of run times for* 30 *experiments with symmetric assignment problems on a* NeXT 68040. *The number of person nodes is* 2000 *and the degree of each is* 8. *For the easy problems, the arc benefits are drawn from the range* [0, 100]. *For the difficult problems,* 80% *of the arc benefits are drawn from the range* [0, 100] *and* 20% *of the arcs have benefit* 100000. *The codes are as follows*: FR: *Unscaled forward/reverse auction.* SFRk: *Scaled forward/reverse auction with* ε*-reduction factor* k. SFk: *Scaled forward auction with* ε*-reduction factor* k.

on the performance of the auction algorithms. Here 80% of the arcs are drawn from the benefit range [0, 100] and 20% of the arcs have benefit 100000. Such arc benefit distributions are generally considered "difficult" for auction algorithms since they tend to stimulate price wars. As mentioned earlier, forward/reverse auction tends to resolve price wars faster than forward auction, and this advantage is manifested dramatically in the results of Fig. 7 for the difficult problems. It should be noted that, in the difficult problem experiments in Fig. 7, the scaling parameters of forward auction were optimized. This optimization resulted in an improvement of roughly a factor of 6 in run time over the codes with the default scaling parameters given in [Ber91].

Except on artificially constructed examples, we have found the performance of unscaled forward/reverse auction remarkably robust. Indeed, it is only in very special

classes of problems that the performance of this algorithm is significantly hampered by the occurrence of price wars. The paper [Cas92] provides a comprehensive computational study of the performance and the robustness of the forward/reverse algorithms for a variety of problem structures.

5.2. Asymmetric assignment problems. The new forward/reverse auction algorithm for asymmetric one-on-one assignment problems was tested versus the asymmetric version of the JV algorithm. We performed tests with two types of randomly generated problems. For both classes of problems, each person node has 10 incident arcs. However, in the first class of problems, the end nodes of the arcs and the arc benefits were generated in a completely random fashion. Figure 8 gives the running times of the scaled forward reverse auction algorithm, the unscaled auction algorithm, and the JV code for this class of problems. A comparison of this figure with Fig. 3 indicates that this class of problems is relatively "easy" for all methods. In particular, price wars were very infrequent, and the unscaled auction algorithm outperformed its scaled version as well as the JV code by a large margin.

The second class of asymmetric assignment problems was specially designed to create price wars by making some nodes difficult to assign. In particular, we introduced two levels of arc benefits that are different by approximately three orders of magnitude. This kind of bipartite problems is quite typical in many applications where nearly infeasible problems frequently arise, e.g., in target tracking applications where potentially false measurements or tracks cannot be matched to confirmed targets. Figure 9 gives the run times of the various codes versus the number of arcs. It can be seen that the run times of both (scaled and unscaled) auction algorithms again grow almost linearly with the number of arcs, but the scaled auction algorithm outperforms the unscaled one by an almost constant factor of 25. The run time of the JV algorithm grows almost quadratically, as it did for symmetric problems. The performance of scaled auction is significantly better than that of the JV algorithm, but unscaled auction is worse than JV in these experiments. Note, however, that our unscaled auction for asymmetric assignment problems does not involve a reverse portion. The initial object prices in all runs were zero, and as mentioned in § 3, upon termination of the forward

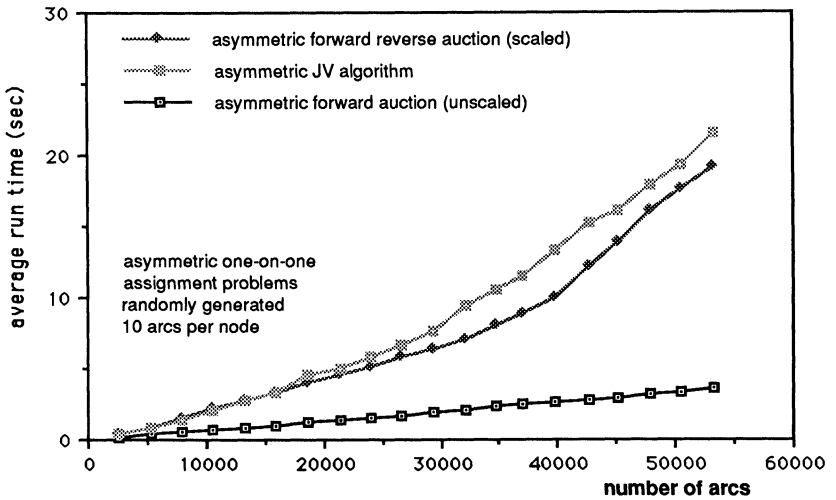

FIG. 8. *Run times for "easy" asymmetric assignment problems on a* MAC II. *The degree of each person node is* 10 *and the arc benefit range is* [0, 1000]. *Each data point represents an average of ten randomly generated problems. The arc benefits are drawn from the benefit range according to a uniform distribution.*

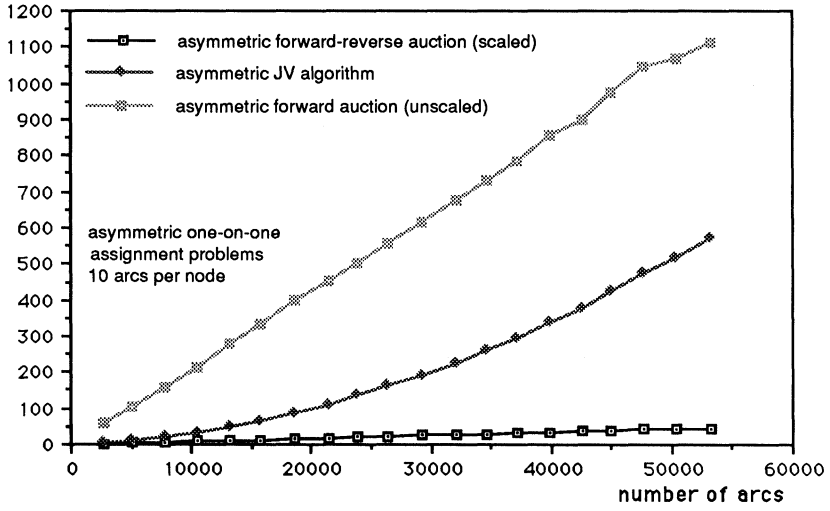

FIG. 9. *Run times for "difficult" asymmetric assignment problems on a* MAC II. *The degree of each person node is* 10. *Each data point represents an average of ten randomly generated problems. The standard deviation for each point was typically less than* 5% *of the corresponding run time. There are two levels of arc benefits that are different by approximately three orders of magnitude.*

auction part of the code, all the ε-CS conditions are satisfied, and the reverse auction part is never used. Thus the mechanism that helped the forward/reverse unscaled auction algorithm to avoid price wars in the difficult problems of Fig. 7 was not employed in the unscaled asymmetric auction algorithm for the difficult problems of Fig. 9.

5.3. Multiassignment problems. Next, we tested the multiassignment auction algorithm (abbreviated *multiauction*) for one-sided asymmetric problems versus the state-of-the-art relaxation code RELAX [BeT88], which solves the equivalent minimum cost network flow problems, and versus the state-of-the-art primal-simplex code NETFLO due to Kennington and Helgason [KeH80], which solves the same equivalent network flow problems. We do not know of any specialized assignment code (including the Hungarian or mixed auction/Hungarian code such as JV) that can be easily modified to handle multiassignment problems. It was found that the NETFLO run times were approximately 40 times higher than those of RELAX for about 5000 arcs and that factor was growing for higher numbers of arcs. In Fig. 10 we show the solution times versus the number of arcs for RELAX and the new multiassignment algorithm for a sequence of randomly generated asymmetric problems with a fixed number of arcs per node. The run times for multiauction grow almost linearly as the number of arcs increases, and this behavior is very consistent across all runs. Furthermore, multiauction is approximately four times faster than RELAX, whose run times also grow roughly linearly but with quite a bit more fluctuation.

Scaling is important for multiassignment problems as it is for one-on-one assignment problems. Although this may not be apparent for randomly generated problems, it is frequently needed in applications, particularly for nearly infeasible problems. In Fig. 11 we show run time results of scaled and unscaled one-sided multiauction algorithms applied to a practical dynamic multi-target tracking and correlation problem over a fixed period of time. At each point in time a sensor's scan produces a set of measurements of target positions that are to be matched with another set of existing

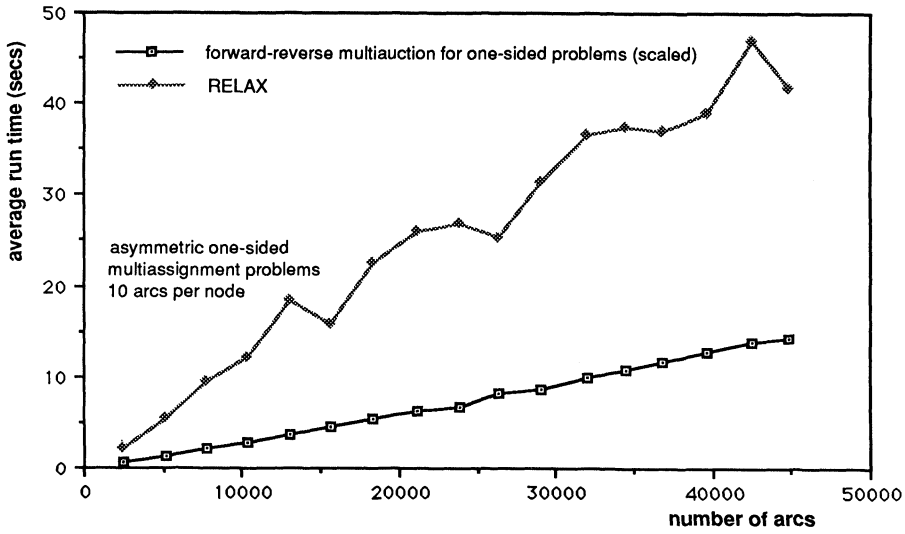

FIG. 10. *Run times for one-sided asymmetric multiassignment problems on a* MAC II. *Each data point represents an average of ten randomly generated problems. The degree of each person node is* 10 *and the arc benefit range is* [0, 1000]. *The standard deviation for each point in the multiauction curve was typically around* 5% *of the corresponding run time, while for the* RELAX *curve it was between* 10 *and* 30%.

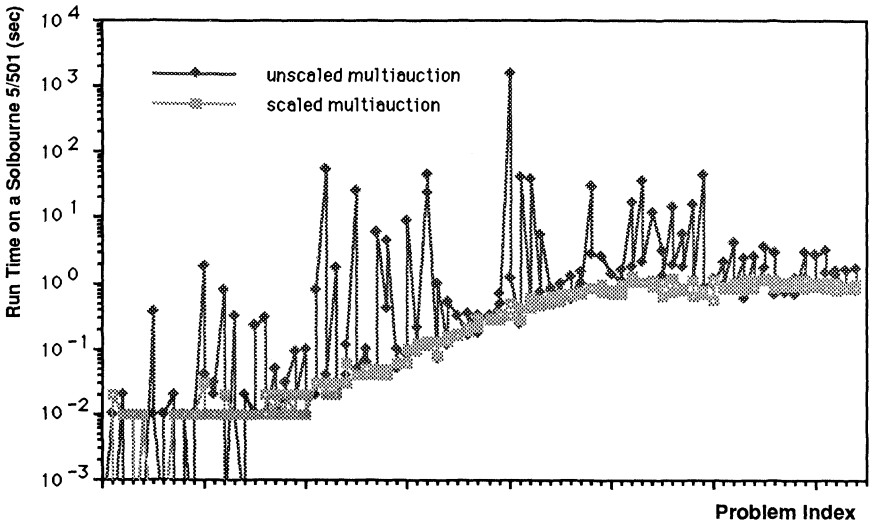

FIG. 11. *Run times for practical one-sided asymmetric multiassignment problems arising in multitarget tracking on a Solbourne* 5/501 *computer, rated at* 22 MIPS. *Each point on the horizontal axis corresponds to a different problem. The number of object nodes ranges from* 10 *to* 125, *and the number of arcs ranges from* 100 *to* 10,000.

tracks by making use of the multiauction algorithms. It can be seen from Fig. 11 that the unscaled multiauction performs worse and far less consistently than the scaled version for this class of practical problems.

Finally, the new two-sided multiassignment algorithm was tested versus RELAX for randomly generated problems. The results, shown in Fig. 12, indicate a substantial speed advantage for the new multiauction algorithm.

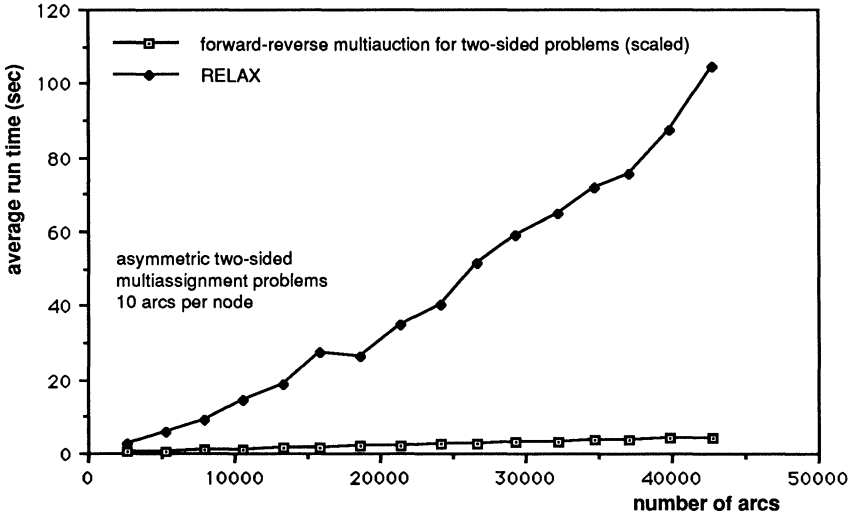

FIG. 12. *Run times for two-sided asymmetric multiassignment problems (cf. (43)) on a* MAC II. *The degree of each person node is* 10 *and the arc benefit range is* [0, 1000]. *In these problems, the upper flow bounds* α_j *are all* ∞.

REFERENCES

[Ber79] D. P. BERTSEKAS, *A distributed algorithm for the assignment problem*, Laboratory for Information and Decision Systems Working Paper, Massachusetts Institute of Technology, Cambridge, MA, March 1979.

[Ber81] ———, *A new algorithm for the assignment problem*, Math. Programming, 21 (1981), pp. 152–171.

[Ber85] ———, *A distributed asynchronous relaxation algorithm for the assignment problem*, Proc. 24th IEEE Conf. Decision and Control, 1985, pp. 1703–1704.

[Ber86a] ———, *Distributed asynchronous relaxation methods for linear network flow problems*, Laboratory for Information and Decision Systems Report LIDS P-1606, Massachusetts Institute of Technology, Cambridge, MA, Nov. 1986.

[Ber86b] ———, *Distributed relaxation methods for linear network flow problems*, Proc. 25th IEEE Conf. Decision and Control, 1986, pp. 2101–2106.

[Ber88] ———, *The auction algorithm: A distributed relaxation method for the assignment problem*, Ann. Oper. Res., 14 (1988), pp. 105–123.

[BeC89a] D. P. BERTSEKAS AND D. A. CASTAÑON *The auction algorithm for transportation problems*, Ann. Oper. Res., 20 (1989), pp. 67–96.

[BeC89b] ———, *The auction algorithm for the minimum cost network flow problem*, Laboratory for Information and Decision Systems Report LIDS-P-1925, Massachusetts Institute of Technology, Cambridge, MA, Nov. 1989.

[BeC89c] ———, *Parallel synchronous and asynchronous implementations of the auction algorithm*, Alphatech Report, Burlington, MA, Nov. 1989; also in Parallel Comput., 17 (1991), pp. 707–732.

[Ber90] D. P. BERTSEKAS, *The auction algorithm for assignment and other network flow problems: A tutorial*, Interfaces, 20 (1990), pp. 133–149.

[Ber91] ———, *Linear Network Optimization: Algorithms and Codes*, M.I.T. Press, Cambridge, MA, 1991.

[Ber92a] ———, *Auction algorithms for network flow problems: A tutorial introduction*, Comput. Optim. Appl., 1 (1992), pp. 7–66.

[Ber92b] ———, *Mathematical equivalence of the auction algorithm for assignment and the ε-relaxation (preflow-push) method for min cost flow*, LIDS Report P-2147, Massachusetts Institute of Technology, Cambridge, MA, Nov. 1992.

[BeE88] D. P. BERTSEKAS and J. ECKSTEIN, *Dual coordinate step methods for linear network flow problems*, Math. Programming, Ser. B, 42 (1988), pp. 203–243.

[BeT85] D. P. BERTSEKAS AND P. TSENG, *Relaxation methods for minimum cost ordinary and generalized network flow problems*, Laboratory for Information and Decision Systems Report LIDS P-1462, Massachusetts Institute of Technology, Cambridge, MA, May 1985; also in Oper. Res. J., 36 (1988), pp. 93–114.

[BeT88] ———, RELAX: *A computer code for minimum cost network flow problems*, Ann. Oper. Res., 13 (1988), pp. 127–190.

[BeT89] D. P. BERTSEKAS AND J. N. TSITSIKLIS, *Parallel and Distributed Computation: Numerical Methods*, Prentice-Hall, Englewood Cliffs, NJ, 1989.

[Bla86] S. S. BLACKMAN, *Multi-target Tracking with Radar Applications*, Artech House, Dedham, MA, 1986.

[CSW89] D. CASTAÑON, B. SMITH, AND A. WILSON, *Performance of parallel assignment algorithms on different multiprocessor architectures*, Alphatech Inc. Report, Burlington, MA, 1989.

[Cas92] D. A. CASTAÑON, *Reverse auction algorithms for assignment problems*, DIMACS Series in Discrete Mathematics and Theoretical Computer Science, 1992.

[Dan63] G. B. DANTZIG, *Linear Programming and Extensions*, Princeton University Press, Princeton, NJ, 1963.

[Gol87] A. V. GOLDBERG, *Efficient graph algorithms for sequential and parallel computers*, Tech. Rep. TR-374, Laboratory for Computer Science, Massachusetts Institute of Technology, Cambridge, MA, Feb., 1987.

[GoT90] A. V. GOLDBERG AND R. E. TARJAN, *Solving minimum cost flow problems by successive approximation*, Math. Oper. Res., 15 (1990), pp. 430–466.

[JoV87] R. JONKER AND A. VOLEGNANT, *A shortest augmenting path algorithm for dense and sparse linear assignment problems*, Computing, 38 (1987), pp. 325–340.

[KKZ89] D. KEMPA, J. KENNINGTON, AND H. ZAKI, *Performance characteristics of the Jacobi and Gauss-Seidel versions of the auction algorithm on the Alliant* FX/8, Report OR-89-008, Dept. of Mechanics and Industrial Engineering, Univ. of Illinois, Urbana, IL, 1989.

[KeH80] J. KENNINGTON AND R. HELGASON, *Algorithms for Network Programming*, John Wiley, New York, 1980.

[PaS82] C. H. PAPADIMITRIOU AND K. STEIGLITZ, *Combinatorial Optimization: Algorithms and Complexity*, Prentice-Hall, Englewood Cliffs, NJ, 1982.

[PhZ88] C. PHILLIPS AND S. A. ZENIOS, *Experiences with large scale network optimization on the connection machine*, Report 88-11-05, Dept. of Decision Sciences, The Wharton School, Univ. of Pennsylvania, Philadelphia, PA, Nov., 1988.

[Roc84] R. T. ROCKAFELLAR, *Network Flows and Monotropic Programming*, Wiley-Interscience, New York, 1984.

[Sch90] B. L. SCHWARTZ, *A computational analysis of the auction algorithm*, unpublished manuscript, 1990.

[WeZ90] J. WEIN AND S. A. ZENIOS, *Massively parallel auction algorithms for the assignment problem*, Proc. Third Symposium on the Frontiers of Massively Parallel Computation, MD, Nov. 1990.

[WeZ91] ———, *On the massively parallel solution of the assignment problem*, J. Parallel Distrib. Comput., 13 (1991), pp. 228–236.

[Zak90] H. ZAKI, *A comparison of two algorithms for the assignment problem*, Report ORL 90-002, Dept. of Mechanical and Industrial Engineering, Univ. of Illinois, Urbana, IL.