Results in Control and Optimization 14 (2024) 100383

Contents lists available at ScienceDirect

Results in Control and Optimization

journal homepage: www.elsevier.com/locate/rico

New auction algorithms for the assignment problem and extensions☆

Dimitri Bertsekas

Fulton Professor of Computational Decision Making, School of Computing, and Augmented Intelligence, Arizona State University, Tempe, AZ, United States of America

ARTICLE INFO

Keywords:
Linear assignment problem
Auction algorithm
Optimal solution
Duality theory

ABSTRACT

We consider the classical linear assignment problem, and we introduce new auction algorithms for its optimal and suboptimal solution. The algorithms are founded on duality theory, and are related to ideas of competitive bidding by persons for objects and the attendant market equilibrium, which underlie real-life auction processes. We distinguish between two fundamentally different types of bidding mechanisms: *aggressive* and *cooperative*. Mathematically, aggressive bidding relies on a notion of approximate coordinate descent in dual space, an ϵ-complementary slackness condition to regulate the amount of descent approximation, and the idea of ϵ-scaling to resolve efficiently the price wars that occur naturally as multiple bidders compete for a smaller number of valuable objects. Cooperative bidding avoids price wars through detection and cooperative resolution of any competitive impasse that involves a group of persons.

We discuss the relations between the aggressive and the cooperative bidding approaches, we derive new algorithms and variations that combine ideas from both of them, and we also make connections with other primal–dual methods, including the Hungarian method. Furthermore, our discussion points the way to algorithmic extensions that apply more broadly to network optimization, including shortest path, max-flow, transportation, and minimum cost flow problems with both linear and convex cost functions.

1. Introduction

In this paper, we discuss auction algorithms for solving numerically the classical assignment (aka weighted bipartite matching) problem, where there are n persons, denoted by $i = 1, \dots, n$, and n objects, denoted by $j = 1, \dots, n$, which we have to match on a one-to-one basis. Each person i may be matched to any object j within a given subset $A(i) \subset \{1, \dots, n\}$. By a *complete assignment* we mean a set of person-object pairs $(1, j_1), \dots, (n, j_n)$, such that $j_i \in A(i)$ for all $i = 1 \dots, n$, while the objects j_i are all distinct. There is a known value a_{ij} for matching person i with object $j \in A(i)$, and we want to find a complete assignment that maximizes the total value

$$\sum_{i=1}^{n} a_{ij_i}.$$

The assignment problem has received a lot of attention since the 1950s. It arises in many practical settings, the most obvious ones being resource allocation problems, such as assigning personnel to jobs, resources to tasks, and related contexts, such as scheduling

☆ Many thanks are due to Yuchao Li for extensive helpful comments.
E-mail address: dpbertsekas@gmail.com.

https://doi.org/10.1016/j.rico.2024.100383
Received 4 January 2024; Accepted 23 January 2024
Available online 26 January 2024

and data association. The assignment problem also appears often as a subproblem in various methods for solving more complex problems.

Recent applications of the assignment problem include:

(a) *Optimal transport* (arising in cosmology among others; see e.g., Brenier et al. [1], Frisch and Sobolevskii [2], Lavaux [3], Villani [4,5], Santambrogio [6], Galichon [7], Metivier et al. [8], Schmitzer [9,10], Walsh and Dieci [11,12], Peyre and Cuturi [13], Levy, Mohayaee, and von Hausegger [14], Merigot and Thibert [15]).
(b) *Graph similarity problems* (arising in computational biology among others; see e.g., Kollias at al. [16], Erciyes [17]).
(c) *Graph neural networks* (see e.g., Zhou et al. [18], Aironi, Cornell, and Squartini [19], Nurlanov, Schmidt, and Bernard [20]),
(d) *Combinatorial auctions* (see e.g., Parkes and Ungar [21], De Vries and Vohra [22]).
(e) *Computational physics* (see e.g., Kosowsky and Yuille [23], Jacobs, Merkurjev, and Esedoglu [24], Bertozzi and Merkurjev [25,26]).
(f) *A variety of dynamic task allocation, scheduling, multiagent, and multi-robot problems* (see e.g., Bayati et al. [27], Bayati, Shah, and Sharma [28], Choi, Brunet, and How [29], Liu and Shell [30], Luo, Chakraborty, and Sycara [31], Morgan et al. [32], Tang et al. [33], Duan et al. [34], Huang, Zhang, and Xiao [35], Luzak et al. [36,37], Otte, Kuhlman, and Sofge [38], Aziz et al. [39], Wang et al. [40], Garces et al. [41], Li et al. [42], and Wang, Li, and Yao [43]).

The assignment problem is also of great theoretical significance because, despite its simplicity, it embodies a fundamental linear programming structure. In particular, the important single commodity linear cost network flow problem can be reduced to an assignment problem by means of a simple reformulation. Thus, any method for solving the assignment problem can be generalized to solve the linear network flow problem, and in fact this approach is particularly helpful in understanding the extensions of auction algorithms to network flow problems that are more general than assignment. Detailed discussions can be found in the author's network optimization textbooks [44,45], and are very relevant to the research directions presented in this paper.

Duality Theory for the Assignment Problem

To develop an intuitive understanding of auction algorithms, it is helpful to introduce an economic equilibrium problem that turns out to be equivalent to the assignment problem. Let us consider the possibility of matching the n objects with the n persons through a market mechanism, viewing each person as an economic agent acting in his/her own best interest. Suppose that object j has a price p_j and that the person who acquires the object must pay the price p_j. Then the (net) profit of object j for person i is $a_{ij} - p_j$ and each person i would logically want to be assigned to a maximal profit object $j_i \in A(i)$, i.e., one satisfying

$$a_{ij_i} - p_{j_i} = \max_{j \in A(i)} \{a_{ij} - p_j\}.$$

A set of prices $p = (p_1, \ldots, p_n)$ and a set of assigned pairs $\mathcal{A} = \{(i_1, j_1), \ldots, (i_k, j_k)\}$ where each assigned person satisfies the preceding condition, i.e.,

$$a_{i_m j_m} - p_{j_m} = \max_{j \in A(i_m)} \{a_{i_m j} - p_j\}, \qquad \text{for all assigned pairs } (i_m, j_m) \in \mathcal{A}, \tag{1}$$

are said to satisfy *complementary slackness* (CS for short). When CS holds, for a set of prices p and a complete assignment $\mathcal{A}$ (i.e., one where $k = n$), we have a form of economic equilibrium whereby each person is assigned to an object that offers maximum profit, and has no incentive to switch to a different object.

A fundamental duality theorem states that a complete assignment that satisfies the CS condition (1) together with some set of prices, is optimal, i.e., it offers maximum total value. Moreover, the corresponding set of prices solves an associated dual optimization problem, which is to minimize over $p = (p_1, \ldots, p_n)$ the dual cost function

$$\sum_{i=1}^{n} \pi_i + \sum_{j=1}^{n} p_j, \tag{2}$$

where π_i is the maximum profit that is attainable for person i under the set of prices p:

$$\pi_i = \max_{j \in A(i)} \{a_{ij} - p_j\}, \qquad i = 1, \ldots, n. \tag{3}$$

Mathematically, we can view this as a consequence of the celebrated duality theorem of linear programming, whereby the assignment optimization is viewed as the primal problem and the minimization of the cost (2)–(3) is the dual problem.[1]

Algorithms for Solving the Assignment Problem

There are several iterative algorithms for the solution of the assignment problem, which are described in detail in several sources, including the linear programming textbook by Bertsimas and Tsitsiklis [46], the network optimization books by Bertsekas [44,45],

[1] The proof is very simple. For any set of prices $(p_1, \ldots, p_n)$ and any complete assignment $(1, j_1), \ldots, (n, j_n)$, using the definition (3) of the profit π_i, we have

$$\sum_{i=1}^{n} \pi_i + \sum_{j=1}^{n} p_j = \sum_{i=1}^{n} \max_{j \in A(i)} \{a_{ij} - p_j\} + \sum_{j=1}^{n} p_j \geq \sum_{i=1}^{n} \{a_{ij_i} - p_{j_i}\} + \sum_{j=1}^{n} p_j = \sum_{i=1}^{n} a_{ij_i}.$$

Under the CS condition (1), equality holds in the above relation. Thus when CS is satisfied, $(p_1, \ldots, p_n)$ attains the minimum of the dual cost on the left side above, while $(1, j_1), \ldots, (n, j_n)$ attains the maximum of the right side.

and Burkard, Dell'Amico, and Martello [47], and the extensive surveys by Ahuja, Magnanti, and Orlin [48,49], and Burkard and Cela [50], among others. In particular, two classical types of methods are:

(a) *Primal simplex methods*, which start with some feasible assignment (a primal basic solution) and iteratively increase the value of the assignment by using the mechanism of the simplex method, suitably adapted to take advantage of the underlying graph structure.
(b) *Dual cost improvement methods*, which include Kuhn's Hungarian method [51], and the relaxation algorithms by Bertsekas [52,53], and Bertsekas and Tseng [54]. These methods start from a dual solution (a set of object prices) and iteratively modify the prices along dual descent directions, thus generating a cost improving sequence of dual solutions.

A third and distinct class of iterative methods for the assignment problem is *auction algorithms*, the subject of this paper. These methods resemble real-life auctions and can be loosely interpreted as approximate coordinate descent methods for solving the dual problem. The approximation is controlled by a parameter $\epsilon > 0$, which may be reduced in the course of the algorithm. Auction algorithms differ from primal methods and dual methods in a fundamental way: they may deteriorate both the primal and the dual objectives at any one iteration by an amount that depends on ϵ. Still, with appropriate implementation and control of the size of ϵ, they find an optimal primal and dual solution pair.

Aims and Contributions of the Paper

The present paper focuses on three related types of auction algorithms, *conservative*, *aggressive*, and *cooperative*, which aim to find a set of prices and a complete assignment that attain the market equilibrium noted earlier, and hence solve the corresponding dual and primal problems. The aggressive auction algorithm was first proposed by the author in the paper [55], and was followed by a proposal of a cooperative auction algorithm in the paper [52]. The conservative auction algorithm, which is a limiting form of the aggressive auction algorithm, was also discussed in these papers, and in fact it was suggested as an effective initialization of some of the cooperative algorithms of the paper [52], despite the fact that in general it does not guarantee convergence to an optimal assignment.[2]

The distinction between the conservative and aggressive auction algorithms can be described in terms of a critical parameter ϵ that characterizes the "intensity" of competition between the persons for the objects: in conservative auction $\epsilon = 0$, while in aggressive auction $\epsilon > 0$. The cooperative auction algorithm, as given in [52], uses $\epsilon = 0$, so it has a conservative character. The present paper extends substantially the cooperative auction framework by allowing $\epsilon > 0$ and by integrating the three different types of auction into a single method, aiming to combine their best characteristics. In particular, the extension to the case where $\epsilon > 0$ involves qualitatively significant changes in the algorithm's character, and appears to be substantially faster for many problems. The new ideas of this paper also point the way towards extensions to network optimization problems that are more general than assignment.

We first review in Section 2 some of the known ideas relating to conservative and aggressive auctions, and the principal challenges that they face due to what we will call *competitive impasses* and *price wars*. In Section 3, we propose a new cooperative auction algorithm, which aims to provide a mechanism for addressing price wars. We discuss several variations, including the *expanding coalitions variant* of cooperative auction, which provides a conceptual vehicle for bridging the ideas of auction and Hungarian methods. The algorithm is structured so that it can combine harmoniously conservative, aggressive, and cooperative auction ideas. A combination of this type was given in the paper [52] for the special case where $\epsilon = 0$. We provide a similar combination, but one where $\epsilon > 0$. In Section 4, we discuss additional variations of the algorithms of Sections 2 and 3, as well as the role of ϵ-scaling within the broader cooperative auction framework of the paper.

In the present paper we will focus on the algorithmic ideas underlying auction algorithms for the assignment problem, particularly the new cooperative versions. In a future report, we will provide results of computational experimentation and describe how our auction ideas can be extended to other linear network flow problems, such as shortest path, max-flow, transportation, and transshipment problems. We will also extend our algorithms of Sections 3 and 4 to single commodity network flow problems with

[2] The term "naive auction" was used instead of "conservative auction" in these and other subsequent works. We will avoid the term "naive" in this paper: it is somewhat misleading because conservative auction embodies interesting ideas, and is useful both conceptually and practically, despite the fact that it does not guarantee convergence to an optimal assignment. The paper [52] also proposed and tested a two-phase algorithm, whereby conservative/naive auction was used in the first phase to initialize a Hungarian algorithm used in the second phase. The code of Jonker and Volgenant [56], often referred to as the "JV code", is very similar. It uses the conservative auction algorithm to initialize a Hungarian-like sequential shortest path method, but starts conservative auction with the classical choice for initial prices: p_j is set to $\min_i a_{ij}$, rather than $p_j = 0$, the author's choice in the code of [52] (in fact the authors of [56] developed their code working from a printout of the author's 1981 code). The JV code has been used widely, as it clearly performs better than codes based on the classical Hungarian method, thanks to its conservative auction initialization. On the other hand, aggressive auction codes seem to outperform the JV code, and other Hungarian-related codes, for many types of problems, although assessments differ on this issue; see e.g., Bertsekas and Eckstein [57], Castañon [58], Zaki [59], Malkoff [60]. Aggressive auction codes also seem to outperform codes that are inspired by preflow-push ideas (whose mechanism can be viewed as mathematically equivalent to the one of the aggressive auction algorithm); see the papers by Bertsekas [61], Naparstek and Leshem [62], Alfaro et al. [63], and the textbook [45] (Section 7.3.3).

separable convex cost functions, building on auction algorithmic ideas presented in the papers by Bertsekas, Polymenakos, and Tseng [64,65], and discussed in more detail in Chapter 9 of the book [45].

2. Conservative and aggressive auctions

Let us first establish some terminology. In what follows, by an *assignment* we mean a set of person-object pairs $(i_1, j_1), \ldots, (i_k, j_k)$, such that $j_1 \in A(i_1), \ldots, j_k \in A(i_k)$, while $i_1, \ldots, i_k$ are distinct persons and $j_1, \ldots, j_k$ are distinct objects. If $k = n$ the assignment is called *complete*, and if $k < n$ the assignment is called *partial* (or *incomplete*). The empty assignment, where there are no assigned persons or objects, is also considered to be a partial assignment. Generally, assignments (complete, incomplete, or empty) will be denoted by $\mathcal{A}$. We assume throughout that there exists at least one complete assignment for our given problem. Also for simplicity in describing algorithms, and without loss of generality, we assume that $A(i)$, the set of objects to which person i can be assigned, contains at least two elements.

A common characteristic of all auction algorithms is that they maintain at all times a partial assignment $\mathcal{A}$ and a set of object prices $p = (p_1, \ldots, p_n)$, which satisfy an approximate form of the CS condition (1) that involves a parameter $\epsilon \geq 0$. The partial assignment grows progressively to become a complete assignment, at which time the auction algorithm terminates.

The central mechanism of an auction algorithm is a bid by an unassigned person i for his/her "best" object j_i (the one that maximizes the person's profit):

$$a_{ij_i} - p_{j_i} = \max_{j \in A(i)} \{a_{ij} - p_j\}.$$

In particular, person i bids for j_i by raising its price from p_{j_i} to $\overline{p}_{j_i}$ given by

$$\overline{p}_{j_i} = \epsilon + \text{the price level that makes the profit of } j_i \text{ equal to the second best profit } \max_{j \in A(i),\, j \neq j_i} \{a_{ij} - p_j\}.$$

Depending on whether $\epsilon = 0$ or $\epsilon > 0$, the auction is called *conservative* or *aggressive*, respectively. Thus in an aggressive auction the object prices are raised by larger increments. In this section we will review these two different types of auction and their properties. For detailed discussions, which include additional topics, such as parallel and asynchronous distributed implementations, we refer to the textbooks [44,66], and [45], and the tutorial papers [67,68]. No new research is presented in this section.

2.1. Conservative auction

As in real-life auctions, a person needs to balance two competing considerations when determining a proper bid size: a high bid for his/her preferred object discourages bids of other persons for that object, but also diminishes his/her profit upon acquiring that object. Thus it makes sense for a person i to maximize the bid for a preferred object j_i subject to the constraint that this object continues to offer maximum profit, i.e., to raise the price of j_i to

$$\overline{p}_{j_i} = a_{ij_i} - w_i,$$

where w_i is the "second best" profit,

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

thereby bringing the profit

$$a_{ij_i} - \overline{p}_{j_i},$$

of the best object j_i to the level of the profit w_i of the second best object; see Fig. 1. We view this auction mechanism as conservative because when selecting a bid, person i takes no risk, in the sense that he/she will never end up with a non-maximum profit object.[3]

Let us describe the conservative auction algorithm more precisely. The algorithm proceeds in iterations and throughout its operation, maintains a set of prices $p = (p_1, \ldots, p_n)$ and a partial assignment $\mathcal{A}$ where each assigned person is assigned to a maximal profit object, i.e., the CS condition (1) is satisfied. It terminates when following an iteration, the assignment obtained is complete. The algorithm starts with any set of prices and partial assignment that satisfy CS; for example it may start with an arbitrary set of prices and the empty assignment. Given the current set of object prices p and partial assignment $\mathcal{A}$, a conservative auction iteration generates a new set of prices and a new assignment as follows.

[3] Price rises below the maximum level $a_{ij_i} - w_i$ also have this property, but larger price rises tend to accelerate the termination of the auction, and are therefore better suited for our algorithmic purposes.

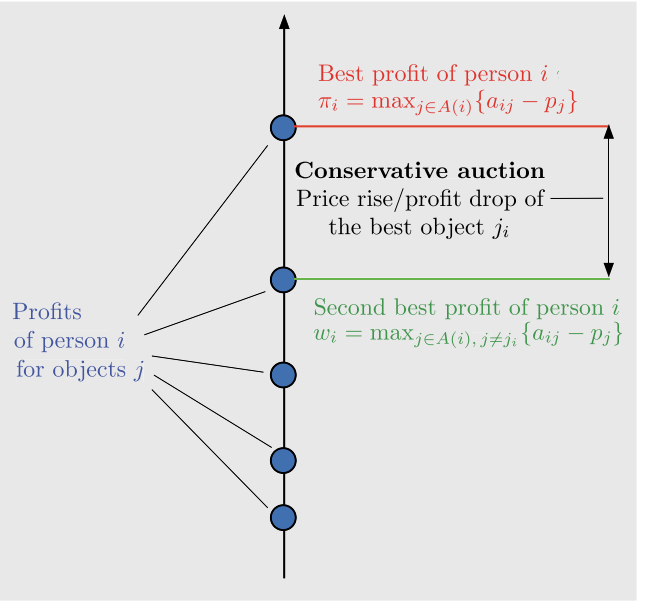


Fig. 1. Illustration of the price rise of the best object j_i of an unassigned person i in the conservative auction algorithm. The price of j_i is increased by $\pi_i - w_i$, while the profit of j_i is made equal to w_i.

Conservative Auction Iteration

We select an unassigned person i and an object j_i that offers maximum profit for i under the given prices,

$$a_{ij_i} - p_{j_i} = \max_{j \in A(i)} \{a_{ij} - p_j\}. \tag{4}$$

We set the price of j_i to

$$\overline{p}_{j_i} = a_{ij_i} - w_i, \tag{5}$$

where w_i is the "second best" profit,

$$w_i = \max_{j \in A(i),\, j \neq j_i} \{a_{ij} - p_j\}. \tag{6}$$

Finally, we add to the assignment $\mathcal{A}$ the pair (i, j_i), and if j_i was assigned to some other person $\tilde{i}$, we remove from $\mathcal{A}$ the pair $(\tilde{i}, j_i)$, thus forming a new assignment $\bar{\mathcal{A}}$.

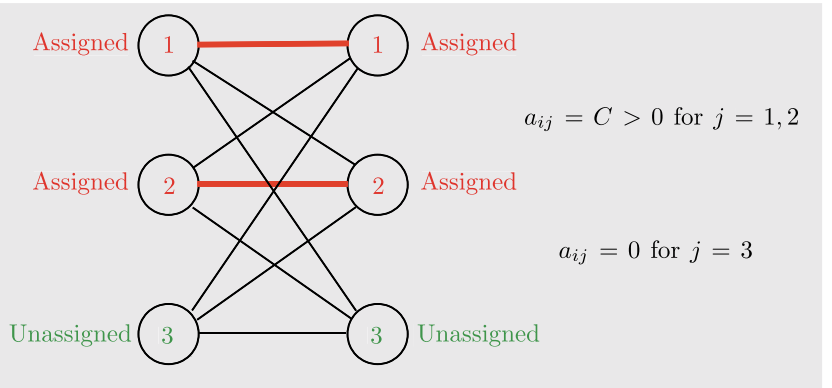
It can be seen that the conservative auction algorithm maintains the CS condition (1) throughout its operation, and generates a sequence of partial assignments whose cardinalities are not decreasing, so if it terminates, the complete assignment obtained at termination is optimal, while the corresponding final prices are an optimal solution to the dual problem, by the duality theorem noted earlier.

On the other hand, conservative auction offers no guarantee of termination: we may end up with a situation where the object prices stop changing, while the cardinality of the current assignment stops growing, as some persons simply change their assigned objects in some way. In particular, by Eqs. (7)–(6), the new price $\overline{p}_{j_i}$ of the preferred object j_i cannot decrease, i.e.,

$$\overline{p}_{j_i} \geq p_{j_i},$$

and it will increase strictly (i.e., $\overline{p}_{j_i} > p_{j_i}$) if and only if the profit $a_{ij_i} - p_{j_i}$ of j_i is strictly larger than the second best profit w_i; cf. Fig. 1. Thus neither the object prices nor the cardinality of the current assignment will change if there are multiple objects that offer maximum profit for person i, and all of these objects are assigned.

Typically, the cause of nontermination of conservative auction can be traced to what we will call a *competitive impasse*. We can somewhat loosely describe competitive impasse as a situation where there is a set of persons that compete for a smaller number of (more than one) equally desirable objects, and there is no apparent way to allocate objects to persons without leaving some person(s) dissatisfied in the end; see the example of Fig. 2. In practice, however, conservative auction can quickly succeed in assigning a substantial number of objects, and for this reason it can be used for effective initialization of other assignment algorithms, as was noted in the papers [52] and [56].



At Start of Iteration #	Object Prices	Assigned Pairs	Bidding Person	Best Object	Bidding Increment
1	$(0, 0, 0)$	$(1, 1), (2, 2)$	3	2	0
2	$(0, 0, 0)$	$(1, 1), (3, 2)$	2	2	0
3	$(0, 0, 0)$	$(1, 1), (2, 2)$	3	2	0

Fig. 2. Illustration of how the conservative auction algorithm may never terminate for a 3×3 assignment problem. Here objects 1 and 2 have value $C > 0$ for all persons, and object 3 has value 0 for all persons. The algorithm starts from the initial prices $p = (0, 0, 0)$ and the partial assignment $\{(1, 1), (2, 2)\}$. There is a competitive impasse involving persons 1, 2, and 3, and objects 1 and 2. The algorithm cycles as persons 2 and 3 alternately bid for object 2 (or object 1) without changing its price because they prefer equally object 1 and object 2.

2.2. Aggressive auction

The aggressive auction algorithm is similar to its conservative counterpart, but guarantees convergence to a complete assignment. In particular, a competitive impasse is resolved by requiring that a bid by an unassigned person i for the best object j_i increases the price of j_i by at least some positive increment ϵ. In particular, person i raises the price of the best object j_i by the amount

$$\pi_i - w_i + \epsilon,$$

where $\pi_i = \max_{j \in A(i)} \{a_{ij} - p_j\}$ is the profit of the best object, given by Eq. (3), and w_i is the second best profit, given by Eq. (6); see Fig. 3. We refer to this type of auction as *aggressive*, because in contrast to the conservative type, it is guaranteed to apply positive price rises (at least ϵ), and it may produce a complete assignment where some of the persons are assigned to a non-maximum profit object. We will also contrast aggressive auction with the cooperative type of auction algorithm (to be discussed shortly), which aims to first detect a competitive impasse and then resolve it through a process of mutual agreement among the competing persons.

In summary, given a set of object prices $(p_1, \ldots, p_n)$ and a partial assignment $\mathcal{A}$, an aggressive auction iteration generates a new set of prices and a new assignment as described below. The algorithm terminates when following an iteration, the assignment obtained is complete.

Aggressive Auction Iteration

We select an unassigned person i and an object j_i that offers maximum profit for i under the given prices,

$$a_{ij_i} - p_{j_i} = \max_{j \in A(i)} \{a_{ij} - p_j\}. \tag{7}$$

We set the price of j_i to

$$\overline{p}_{j_i} = a_{ij_i} - w_i + \epsilon \tag{8}$$

where w_i is the "second best" profit,

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

Finally, we add to the assignment $\mathcal{A}$ the pair (i, j_i), and if j_i was assigned to some other person $\tilde{i}$, we remove from $\mathcal{A}$ the pair $(\tilde{i}, j_i)$, thus forming a new assignment $\bar{\mathcal{A}}$.

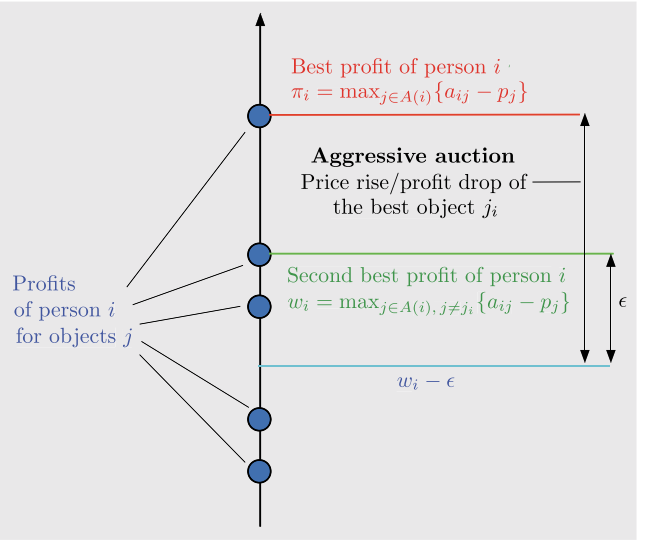


Fig. 3. Illustration of the price rise of the best object j_i of an unassigned person i in the aggressive auction algorithm. The price of j_i is increased by $\pi_i - w_i + \epsilon$, while the profit of j_i is strictly decreased to $w_i - \epsilon$.

It can be shown that the algorithm is guaranteed to terminate (under our assumption that there exists at least one complete assignment; see the original paper [55], or the books [44,45,66] for a proof). Intuitively, the reason is that each bid by a person i is guaranteed to increase the price of his/her best object j_i by at least the positive increment ϵ, thus making j_i "less attractive" for other persons. If the auction did not terminate, the prices of the assigned objects would eventually increase to sufficiently high levels to make some of the unassigned objects attractive enough to receive bids and join the assignment. This is similar to what happens in real-life auctions.

The aggressive auction algorithm is designed to maintain the following relaxed form of the CS condition (1), called ϵ-*complementary slackness* (ϵ-CS for short):

$$a_{ij_i} - p_{j_i} \geq \max_{j \in A(i)} \{a_{ij} - p_j\} - \epsilon, \qquad \text{for all assigned pairs } (i, j_i), \tag{9}$$

provided the initial set of prices and partial assignment satisfy this condition. One possibility to satisfy the ϵ-CS condition initially is to start with an arbitrary set of prices and the empty assignment. There are also other more sophisticated possibilities for selecting favorable initial conditions.

Thanks to the ϵ-CS condition, it can be shown that the final assignment obtained is optimal within $n\epsilon$, and hence exactly optimal if the values a_{ij} are integers and $\epsilon < 1/n$. To see this, note that the complete assignment and set of prices obtained at termination *satisfy CS for a fictitious/slightly perturbed problem* where all values a_{ij} are the same as before, except for the values a_{ij_i} of the n assigned pairs (i, j_i), which are modified by an amount of no more than ϵ; cf. Eq. (9). The final complete assignment is optimal for this perturbed problem, and therefore also optimal within $n\epsilon$ for the original (unperturbed) problem. Thus thanks to the extra ϵ bidding increment, the aggressive auction algorithm succeeds in terminating with a complete assignment, at the risk of some persons ending up with a non-maximum profit object (by as much as ϵ), and an attendant error of at most $n\epsilon$ from optimality.

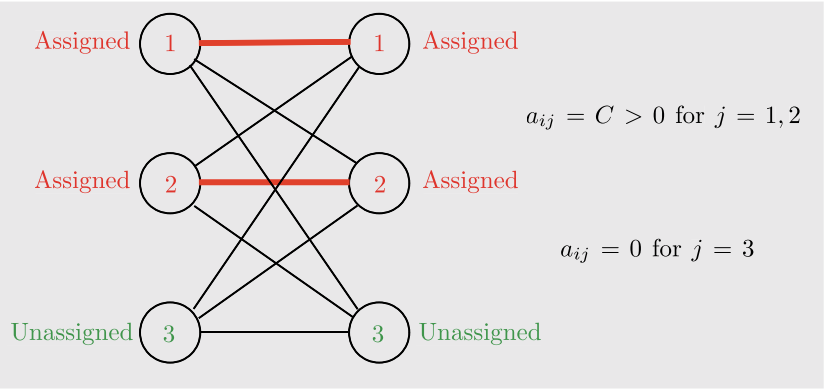
Unfortunately, the aggressive auction algorithm runs into another difficulty, which can also be traced to a competitive impasse. This difficulty, called a *price war*, refers to a protracted sequence of small price rises of order ϵ, which results from groups of persons competing for a smaller number of two or more objects that are more or less equally desirable. An example of a price war in the case of a 3×3 assignment problem is given in Fig. 4, and it can be seen that it degrades computational efficiency. In particular, the number of iterations in this example is proportional to C/ϵ, and a similar example (given as Exercise 7.4b in the book [45]) shows that the number of iterations needed to resolve a price war can be as high as nC/ϵ.

Generally, the complexity of the algorithm can be shown to be proportional to C/ϵ, where

$$C = \max_{i=1,\ldots,n,\, j \in A(i)} |a_{ij}| \tag{10}$$

is the range of object values. Thus, the complexity is pseudopolynomial and is often unacceptable. In actual use of the aggressive auction algorithm, price wars are common, particularly when the range C is large and the assignment problem is sparse, i.e., each person can be assigned to only a small subset of objects).

One way to overcome the detrimental effect of price wars is ϵ-*scaling*, a natural computational idea that was noted in the original aggressive auction proposal of the paper [55]. Here the algorithm is first run for a fairly large initial value of ϵ, to converge quickly and yield good object price estimates. These estimates are used to initialize an aggressive auction with a reduced value of ϵ. After several successive rounds of ϵ-reduction by some constant factor, this process will bring ϵ to a sufficiently low level to produce



At Start of Iteration #	Object Prices	Assigned Pairs	Bidding Person	Best Object	Bidding Increment
1	$(0, 0, 0)$	$(1, 1), (2, 2)$	3	2	ϵ
2	$(0, \epsilon, 0)$	$(1, 1), (3, 2)$	2	1	2ϵ
3	$(2\epsilon, \epsilon, 0)$	$(2, 3), (3, 1)$	1	2	2ϵ
4	$(2\epsilon, 3\epsilon, 0)$	$(1, 2), (2, 1)$	3	1	2ϵ
5	$(4\epsilon, 3\epsilon, 0)$	$(1, 3), (3, 2)$	2	2	2ϵ
6	...	...	...	...	...

Fig. 4. Illustration of how the aggressive auction algorithm overcomes the competitive impasse problem for the 3×3 example of Fig. 2 by making the bidding increment at least equal to ϵ. The table shows one possible sequence of bids and assignments generated by the auction algorithm, starting with all prices equal to 0 and the partial assignment $\{(1, 1), (2, 2)\}$. At each iteration except the last, the unassigned person bids for either object 1 or 2, increasing its price by ϵ in the first iteration and by 2ϵ in each subsequent iteration. In the last iteration, after the prices 1 and 2 rise to or above C, object 3 receives a bid and the auction terminates. The number of iterations for this to happen is roughly C/ϵ.

an optimal assignment. It can be shown that the (worst-case) computational complexity of aggressive auction with ϵ-scaling is polynomial, $O\big(nm \log(nC)\big)$, where m is the number of arcs of the bipartite graph representing the assignment problem and C is the range of values, given by Eq. (10). This estimate was derived in the author's textbook [66] (Section 5.4) and paper [69], following a progression of related complexity analyses for the max-flow and the minimum cost flow problem involving several works (Karzanov [70], Shiloach and Vishkin [71], Goldberg and Tarjan [72,73], Bertsekas and Eckstein [57,74], Ahuja, Magnanti, and Orlin [48,49], Ahuja and Orlin [75], Cheriyan and Maheshvari [76], Orlin and Ahuja [77]). The recent papers by Naparstek and Leshem [62], and Khosla and Anand [78] provide probabilistic complexity analyses.

For an account of the computational complexity aspects of the aggressive auction algorithm with ϵ-scaling, see the textbooks [66] (Section 5.4) and [45] (Section 7.1.2). The latter textbook also contains detailed discussions (including computational complexity) of extensions of the auction algorithm to related problems, such as asymmetric assignment problems, max-flow, minimum cost flow, with both linear (in Chapter 7) and convex separable cost (in Chapter 9).

Aggressive auction with an efficient ϵ-scaling implementation is widely recognized as one of the most effective assignment algorithms.[4] Several code implementations are publicly available, including some (written in FORTRAN and dating from the early 90s) that can be found in the author's website. A recent code, written in MATLAB, has been made available by Bernard [79]. The algorithm typically outperforms its competitors by a wide margin, as has been shown convincingly by many computational studies. Its advantage is particularly pronounced when good initial object price estimates are available. As a result, the method is very efficient in situations where many similar assignment problems are solved with small variations in their data. Then the final

[4] The experimental verification of the advantages of the aggressive auction algorithm took a long time to establish, owing in part to the primitive state of computer technology at the time. Indeed, given that the aggressive auction algorithm appeared to be radically different from the established assignment algorithms in 1979, like primal simplex and Hungarian, and lacking a thorough computational comparison, the author harbored deep doubts about its effectiveness. In fact, these doubts prompted the development of an alternative cooperative algorithm (with $\epsilon = 0$), which appeared to be conceptually closer to the Hungarian method, the most popular assignment algorithm at the time; see [52]. The story of the discovery of the aggressive auction algorithm is recounted near the end of a videolecture by the author that can be found at https://www.youtube.com/watch?v=T-fSmSqzcqE

prices for a given problem solution can be used as starting prices for solution of other similar problems, often with impressive computational savings.[5]

Another advantage of the aggressive auction algorithm and its extensions to other network flow problems is that it is well-suited for parallel computation, and it is valid even when it is implemented as a distributed asynchronous algorithm. This has been established in the book by Bertsekas and Tsitsiklis [66] (Sections 5.3 and 6.5), as well as in several related computational studies: Bertsekas and Castañon [91], Wein and Zenios [92], Amini [93], Bertsekas at al. [94], Beraldi, Guerriero, and Musmanno [95–97], Zavlanos, Spesivtsev, and Pappas [98], Bus and Tvrdk [99], Sathe, Schenk, and Burkhart [100], Nascimento at al. [101], Naparstek and Leshem [62], Sena, Silva, and Nascimento [102]. The cooperative auction algorithms to be discussed next, can also use good initial price estimates with advantage, but they are not as well suited for distributed computation.

3. Cooperative price rises and cooperative auction

We will now consider an alternative approach for dealing with competitive impasses and price wars. The key characteristic that differentiates it from the aggressive auction approach is the use of multiple-object price rises that aim to forestall price wars. In effect, a group of persons recognize that they are caught up in a multi-object competitive impasse, and rather than engage in a time consuming price war, they collectively agree to raise the prices of the relevant objects by a large common increment, thus preparing to bid for additional objects without violating ϵ-CS.

We call such multi-person bid mechanisms *cooperative*, and we will show that they can be combined harmoniously with the aggressive and conservative auction mechanisms that involve bids by a single person. This idea dates to the paper [52], which included combinations of cooperative multi-person bids with conservative single-person bids, and experimentally demonstrated the potential advantages of such combinations.

To understand the cooperative auction mechanism, let us consider the 3×3 assignment problem of Figs. 2 and 4. There, starting with zero prices, persons 1, 2, and 3 compete for valuable objects 1 and 2 (value C), and aim to avoid assignment to the valueless object 3. As we have seen in Fig. 2, conservative auction fails for this problem, due to a competitive impasse created by perpetual zero-increment bids by persons 1, 2, and 3, for the two desirable objects 1 and 2. Aggressive auction succeeds in finding the optimal assignment after a protracted price war that lasts for about C/ϵ iterations, as illustrated in Fig. 4. Cooperative auction, aims instead to detect the competitive impasse, to identify the set of persons that are involved in it, and to *form a coalition of these persons for the purpose of performing a cooperative price rise* to resolve quickly the impasse within the coalition. In particular, persons 1, 2, and 3 agree to raise the prices of objects 1 and 2 from 0 to $C + \epsilon$, preserving ϵ-CS, while allowing object 3 to be assigned at the next iteration, thus resolving the competitive impasse without a price war.[6] This example also illustrates that *price wars involve more than one object*. This motivates the use of an aggressive bidding approach when the ϵ-zone of the bidding person contains only one object. We will return to this theme later in this section.

We will now extend the idea just described to the general $n \times n$ assignment problem. To this end we need to address the following issues:

(a) The algorithm should maintain a partial assignment and a set of prices that satisfy CS (or ϵ-CS). Thus, once the algorithm terminates, the complete assignment obtained at termination is optimal (or optimal within $n\epsilon$, respectively).
(b) As in the case of conservative and aggressive auctions, the algorithm should aim to enlarge the current partial assignment as long as this is done without violating CS or ϵ-CS.
(c) The algorithm needs an explicit or implicit mechanism to detect that there is a competitive impasse or price war going on. It also needs a mechanism to identify the coalition of persons that are involved in the price war; this coalition will involve a single unassigned person and $m > 1$ assigned persons, and the corresponding assigned m objects for which the $m + 1$ persons compete.
(d) Once a coalition of $m+1$ persons is detected, the prices of the corresponding m assigned objects should be raised simultaneously through a cooperative price rise that does not violate CS or ϵ-CS. An efficient mechanism to calculate the cooperative price rise level should be incorporated into the algorithm.

In what follows in this paper, we will aim to design a broad class of algorithms and variations thereof, which are based on the preceding considerations, and mitigate the occurrences of competitive impasses and price wars. To this end we introduce some definitions, all of which refer to a specific set of prices $p = (p_1, \ldots, p_n)$ and partial assignment $\mathcal{A}$ satisfying ϵ-CS for some fixed $\epsilon \geq 0$ (note that $\epsilon = 0$ is a possibility). If $\epsilon > 0$, the algorithm can be combined with ϵ-scaling, i.e., applying the algorithm with larger values of ϵ to obtain good starting prices for applying the algorithm with smaller values of ϵ. However, the algorithm works even with $\epsilon = 0$.

Preliminary Concepts

We first introduce the concept of ϵ-zone of a person, a new idea that plays a central role in this paper.

[5] Such situations arise often in practice. An example is data association contexts, where related two-dimensional assignment problems are solved repeatedly; see the author's monograph [80] (Section 3.4.2) and paper [81], and references on multi-target tracking, such as Blackman [82], Bar-Shalom and Fortman [83], Bar-Shalom [84], Castañon [85], Pattipati, Deb, Bar-Shalom, and Washburn [86], Poore [87], Poore and Robertson [88], Popp, Pattipati, and Bar-Shalom [89], and Emami et al. [90].

[6] In a real auction the person that is ultimately assigned to the valueless object 3 may need to be compensated by prior agreement with his/her coalition partners. This issue is not addressed in this paper, because our objective is computational efficiency in solving the assignment problem, and not the design of fair real-life auction mechanisms. Some possibilities include consideration of profit sharing between persons, or randomized solutions, whereby persons can acquire fractional amounts of multiple objects, with the fractions adding to 1 for each person.

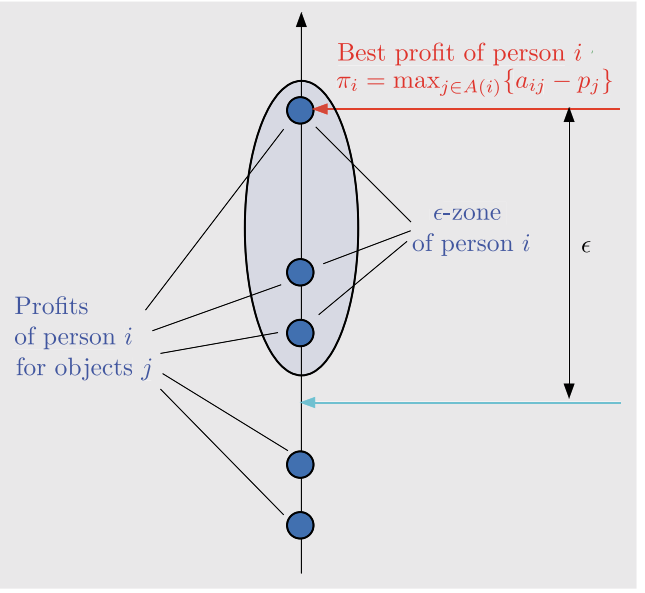


Fig. 5. Illustration of the ϵ-zone of a person i. It consists of all the objects whose profit is within ϵ of the maximum profit $\pi_i = \max_{j \in A(i)} \{a_{ij} - p_j\}$.

Definition 3.1 (*ϵ-Zone of a Person*)**.** Given a set of prices p, the *maximum profit* of a person i, denoted π_i, is defined as

$$\pi_i = \max_{j \in A(i)} \{a_{ij} - p_j\}.$$

For a given $\epsilon \geq 0$, the *ϵ-zone of a person i*, denoted $\mathcal{Z}(i)$, is the set of objects j whose profit for i is within ϵ of being maximal:

$$\mathcal{Z}(i) = \left\{ j \in A(i) \mid a_{ij} - p_j \geq \pi_i - \epsilon \right\}.$$

Fig. 5 illustrates the above definition. Note that the ϵ-zone $\mathcal{Z}(i)$, roughly speaking, consists of the "almost best" objects of person i (those whose profit is within ϵ of being best). It always contains the maximum profit object(s) for person i (and only those if $\epsilon = 0$). Moreover, if a person i is assigned to an object j while ϵ-CS holds, then j belongs to the ϵ-zone $\mathcal{Z}(i)$.

Definition 3.2 (*Alternating Path*)**.** Let a set of prices p and a partial assignment $\mathcal{A}$ satisfying ϵ-CS be given. An *alternating path* is a person sequence $(i, i_1, \ldots, i_k)$ and corresponding object sequence $(j_1, \ldots, j_k)$, $k \geq 1$, such that:

(a) The person i is unassigned, while the persons $i_1, \ldots, i_k$ are assigned to objects $j_1, \ldots, j_k$, respectively.
(b) The object j_1 belongs to the ϵ-zone of person i, while for $m = 2, \ldots, k$, the object j_m belongs to the ϵ-zone of person i_{m-1}.

Definition 3.3 (*Augmenting Path*)**.** Let a set of prices p and a partial assignment $\mathcal{A}$ satisfying ϵ-CS be given. An *augmenting path* is an alternating path $(i, i_1, \ldots, i_k)$, as per Definition 3.2, together with an unassigned object j that belongs to the ϵ-zone of i_k. Given such a path, a corresponding *augmentation* consists of assigning person i to j_1, reassigning person i_k to j, and reassigning persons $i_1, \ldots, i_{k-1}$ to objects $j_2, \ldots, j_k$, respectively (thereby increasing the cardinality of the assignment by one, while maintaining ϵ-CS). Assigning an unassigned person i to an unassigned object j within his/her ϵ-zone $\mathcal{Z}(i)$ is also viewed as an augmentation.

An augmenting path as defined above, is denoted by $(i, i_1, \ldots, i_k, j)$, while the corresponding alternating path is denoted by $(i, i_1, \ldots, i_k)$ [in the case where i is assigned to j, the augmenting path is denoted (i, j)]. Figs. 6 and 7 illustrate alternating and augmenting paths. Key observations here are that:

(a) An augmenting path starts with an unassigned person i and ends with an unassigned object j, while all other persons and objects in the path are assigned.
(b) Person i and object j can get assigned through an augmentation, which reassigns objects to persons, while maintaining ϵ-CS. This augmentation makes progress towards obtaining a complete assignment.

The concepts of alternating and augmenting paths are well-known (for the case $\epsilon = 0$) in the theory of assignment, matching, and max-flow algorithms. In particular, an augmentation increases the cardinality of the assignment by 1, while changing the maximal profits of the persons of the augmenting path by no more than ϵ. Thus an augmentation makes intuitive sense for small values of ϵ.

We now introduce a notion of coalition of persons, which is central in cooperative auction.

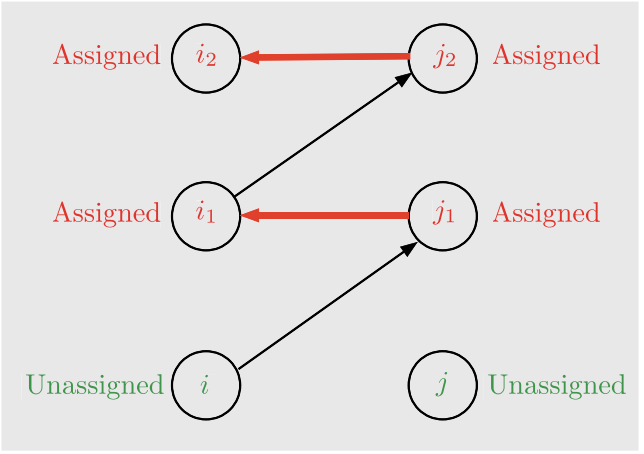


Fig. 6. Illustration of an alternating path (i, i_1, i_2), consisting of the three persons in a 3×3 assignment graph. The first person is unassigned and the subsequent persons are assigned. The objects in the alternating path must belong to the ϵ-zones of the corresponding persons in the path, i.e., $j_1 \in \mathcal{Z}(i)$ and $j_2 \in Z(i_1)$ [in addition to $j_1 \in Z(i_1)$ and $j_2 \in Z(i_2)$, which is true by ϵ-CS]. Another alternating path is (i, i_1).

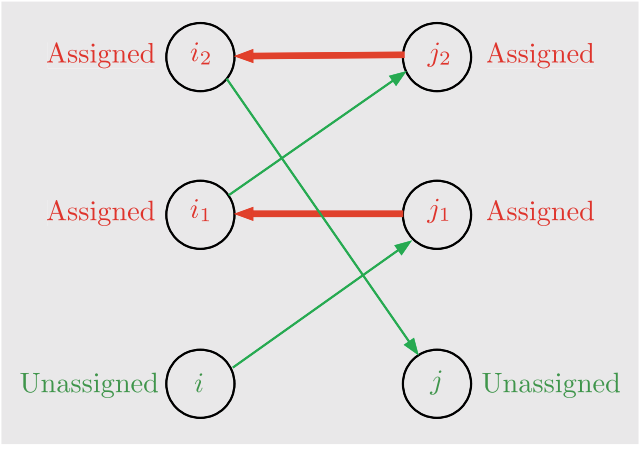


Fig. 7. Illustration of an augmenting path (i, i_1, i_2, j) in a 3×3 assignment graph. It consists of the alternating path (i, i_1, i_2) (cf. Fig. 6), followed by the unassigned object j, which belongs to the ϵ-zone of person i_2.

Definition 3.4 (*Coalition Partners of an Unassigned Person*)**.** Let a set of prices p and a partial assignment $\mathcal{A}$ satisfying ϵ-CS be given, and let i be an unassigned person. A person i' is said to be a *coalition partner* of i if there is an alternating path that starts with i and ends with i'. The set of persons consisting of person i together with all his/her coalition partners is called the *coalition of* i and is denoted by $C(i)$.

Fig. 8 provides illustrations of $C(i)$, the coalition of i. Generally, $C(i)$ consists of a single unassigned person, namely i, together with $m \geq 0$ assigned coalition partners. It consists of the single person i [$C(i) = \{i\}$] if and only if the ϵ-zone of i does not include any assigned objects (cf. the top left graph of Fig. 8). We note that we can obtain $C(i)$ by using a form of forward search that progressively generates a tree of alternating paths starting from i, until no more assigned persons can be found; see the implementation details given later in this section.

Cooperative Auction Iteration

An auction iteration involving a cooperative price rise can now be described in words. We are given a set of object prices $p = (p_1, \ldots, p_n)$ and a partial assignment $\mathcal{A}$ satisfying ϵ-CS. The iteration starts with an unassigned person i and tries to generate $C(i)$, the coalition of i. When $C(i)$ is obtained without intermediate discovery of an augmenting path, the prices of all the objects involved in the coalition will be simultaneously raised. Similar to the aggressive auction iteration, the price rise amount exceeds ϵ, and is the maximum possible that preserves ϵ-CS.

We will now state in detail the iteration just described in summary; see the block diagram of Fig. 9. We may call this iteration "purely" cooperative, to distinguish it from a method that involves a combination with the conservative and aggressive iterations. We will describe this combined method later in this section.

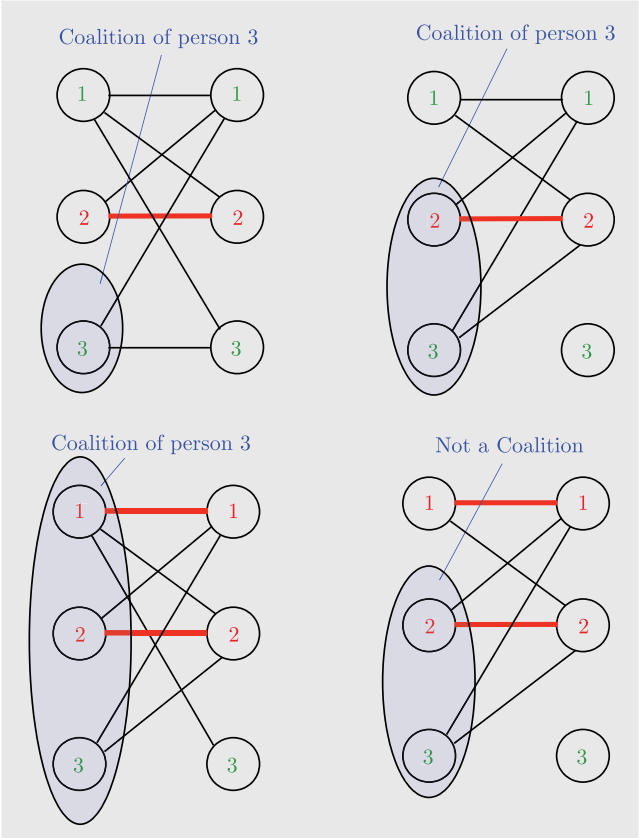


Fig. 8. Illustrations of different cases of $C(3)$, the coalition of the unassigned person 3 in a 3×3 assignment problem. In each of the four cases, an arc (i, j) indicates membership of object i in the ϵ-zone of person i (other arcs are not shown). Red arcs correspond to assigned pairs, black arcs to unassigned pairs.

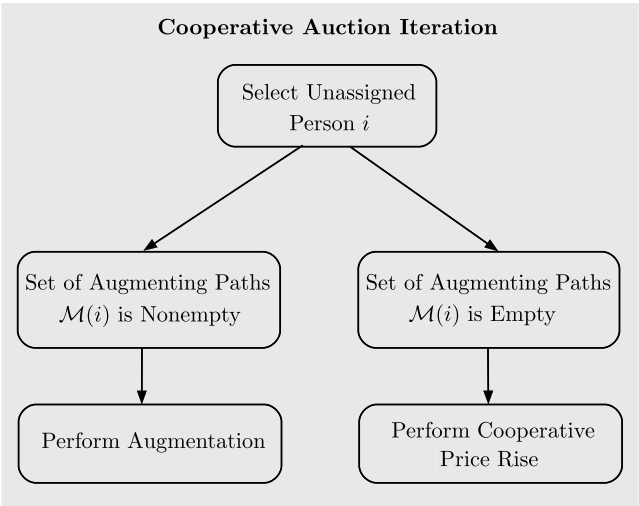


Fig. 9. Block diagram of a cooperative auction iteration.

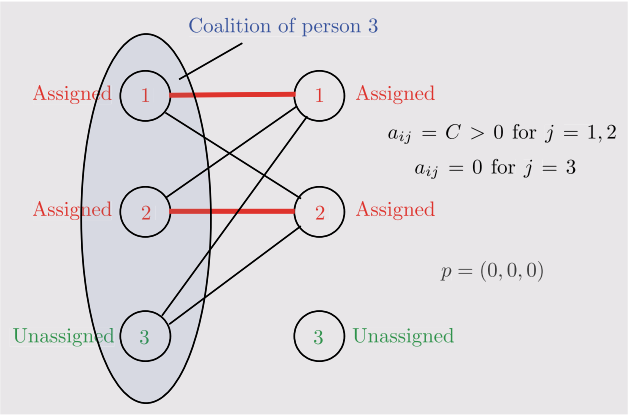


Fig. 10. Illustration of the cooperative auction iteration for the example of Figs. 8, assuming that $C > \epsilon$. Here the coalition partners of the unassigned person 3 are the persons 1 and 2. The cooperative auction iteration consists of a price rise of objects 1 and 2 from 0 to $C + \epsilon$, followed by object 3 coming into the ϵ-zones of all the persons, and allowing an augmentation along $(3, 3)$ that completes the assignment.

Cooperative Auction Iteration

Given a set of object prices $p = (p_1, \ldots, p_n)$ and a partial assignment $\mathcal{A}$ satisfying ϵ-CS, select an unassigned person i. Let $\mathcal{M}(i)$ be the set of augmenting paths that start with i.

- If $\mathcal{M}(i)$ is nonempty, perform an augmentation along some augmenting path from $\mathcal{M}(i)$, increase the price of the last object in this augmenting path by the maximum amount that will not violate ϵ-CS, and go to the next iteration.
- If $\mathcal{M}(i)$ is empty, let $\mathcal{O}(i)$ denote the set of objects that are assigned to some coalition partner of i. Raise the prices of the objects in $\mathcal{O}(i)$ by the maximum common amount for which the ϵ-zone of every person i' in $C(i)$ is a subset of the ϵ-zone of the same person i' after the price rise.

The preceding iteration description of the cooperative auction algorithm leaves out the details of the computations of the sets $\mathcal{M}(i)$, $C(i)$, and $\mathcal{O}(i)$, and the price rise amount. To implement efficiently the iteration, it is necessary to properly organize and streamline these computations. The data structures and procedures for doing so are similar to well-known implementations of auction, Hungarian, and dual descent algorithms, and will be discussed later (cf. Sections 3.3 and 3.4).

Let us illustrate the steps of the cooperative auction iteration with an example.

Example 3.1. Consider the 3×3 assignment example of Figs. 2 and 4. We will describe a single iteration of the cooperative auction algorithm, starting with set of prices $p = (0, 0, 0)$ and partial assignment $\{(1, 1), (2, 2)\}$.

The iteration starts with person 3, the only one left unassigned. We assume that $C > \epsilon$, so the ϵ-zone $Z(3)$ is the set of objects $\{1, 2\}$. Thus we need to construct the coalition of person 3 with a view towards a cooperative price rise. The alternating paths are $(3, 1)$, $(3, 2)$, $(3, 1, 2)$, and $(3, 2, 1)$, so the coalition partners of person 3 are persons 1 and 2, as illustrated in Fig. 10. No augmenting path can be found, i.e., $\mathcal{M}(3)$ is empty, so we increase the prices of objects 1 and 2 by the maximum amount that will not violate ϵ-CS. Thus the prices of objects 1 and 2 are raised to $C+\epsilon$, adding object 3 to the ϵ-zones of persons 1, 2, and 3. At the next iteration, the augmenting path $(3, 3)$ will be discovered, and the algorithm will terminate with an augmentation along $(3, 3)$.

Thus the cooperative auction algorithm terminates very quickly in this example. By contrast, the conservative auction algorithm would not terminate at all because of a competitive impasse (cf. Fig. 2), while the aggressive auction algorithm would require about C/ϵ iterations because of a price war (cf. Fig. 4).

Variants and Modifications

Note that in the preceding example, the augmenting path $(3, 3)$ is created immediately following the price rise, so the corresponding augmentation can be done right away. This suggests a modification of the cooperative auction algorithm so that when an augmenting path is discovered following a price rise, the corresponding augmentation is done right away, rather than wait for another iteration. The expanding coalition variant of the algorithm, which will be discussed shortly, embodies this modification.

If on the other hand an augmenting path is not discovered immediately following a cooperative price rise, there is also a possibility to assign person i through a reassignment of the coalition partners of i. We can view this as a somewhat more aggressive form of collective bid of the coalition $C(i)$, which aims to acquire a new object for the coalition, at the expense of deassigning a person from outside the coalition. It leads to the person reassignment variant of the cooperative auction algorithm, which will be discussed in Section 4.

Similarities with Noncooperative Auction Iterations

Some similarities with the noncooperative auction iterations, which suggest interesting algorithmic variants, are noteworthy. In particular, assume that the ϵ-zone $\mathcal{Z}(i)$ contains a *single unassigned* object (by necessity the maximum profit object). Then the cooperative auction iteration will produce identical results with the conservative iteration (if $\epsilon = 0$) and with the aggressive iteration (if $\epsilon > 0$): it will assign i to that object and raise its price by an amount that exceeds ϵ. If on the other hand the ϵ-zone $\mathcal{Z}(i)$ contains a *single assigned* object j, the results will be different, because the person assigned to j is a coalition partner of i, and this will trigger the mechanism for computing and enlarging the coalition of i.

In what follows (Section 3.2), we will discuss a variant of the cooperative algorithm that behaves identically with the aggressive auction iteration when $\mathcal{Z}(i)$ contains a single object (assigned or unassigned), and is much faster, both in theory and in practice. The motivation for this variant is that the aggressive auction iteration is known to work very fast in the absence of price wars, which involve competition for multiple objects, so *a potential price war is not an issue when $\mathcal{Z}(i)$ consists of a single object*.[7] In Section 4 we will describe still another variant of the cooperative algorithm, which behaves identically with the aggressive auction iteration when there is at most one coalition partner of i [rather than $\mathcal{Z}(i)$ containing a single object]. This is the variant noted earlier, which involves reassignment of the coalition partners of i immediately following a cooperative price rise.

3.1. Cooperative auction iteration with coalition expansions

When the augmenting path set $\mathcal{M}(i)$ is empty (which will happen when all coalition partners of i are assigned), the initial unassigned person i will remain unassigned at the end of the iteration. In this case, since the choice of the unassigned person to start the next iteration is unrestricted, we have the option to start with the same person i. Then the new set of coalition partners of i will include the preceding set of coalition partners, so the *coalition $C(i)$ will be simply expanded and need not be rebuilt from scratch* [by design the ϵ-zone of every person i' in $C(i)$ before the price rise is a subset of the ϵ-zone of the same person i' after the price rise].

This observation motivates an interesting variant of the cooperative auction iteration, which involves multiple successive coalition expansions started by the same single person i, up to the point where an augmentation takes place; see Fig. 11. We call this the *expanding coalitions variant*, and we note that it will always terminate with an augmentation, resulting in assignment of the starting person i, and an increase of the cardinality of the current assignment by 1. Thus, it will produce a complete assignment in exactly n iterations, starting from an empty assignment, while maintaining ϵ-CS throughout the process (under our assumption that the problem is feasible so a complete assignment exists).

Example 3.2. To illustrate the coalition expansion process, let us consider a 4×4 version of the 3×3 problem of Figs. 2 and 4. Here, in addition to the three persons and objects of these figures, there are a fourth person 4 and object 4, as shown in Fig. 12. Person 4 can be assigned to object 3 with value 0 and to object 4 with value -1. Every feasible assignment must include the pair $(4,4)$, so the optimal assignments are the ones of the 3×3 problem, augmented with $(4,4)$, such as for example

$$\big\{(1,1),(2,2),(3,3),(4,4)\big\}. \tag{11}$$

Let the initial prices be $p = (0,0,0,0)$ and the initial partial assignment be

$$\big\{(1,1),(2,2),(4,3)\big\},$$

as shown in Fig. 12. The cooperative auction iteration starts with the unassigned person 3, and constructs the coalition $C(3) = \{1,2,3\}$. In the expanding coalition variant, the search for coalition partners continues after the price rise of objects 1 and 2 (by the amount $C + \epsilon$), adding person 4 to the coalition, which brings object 4 into the ϵ-zone of person 4, and allows the augmentation $(3,4,4)$ and termination with the assignment (11).

Here is a more complicated example, which also demonstrates the potentially significant computational savings for reusing the computation of previous coalitions to save in the computation of subsequent coalitions.

Example 3.3 (*Computational Advantage of Expanding Coalitions*)**.** Consider the $n \times n$ assignment example of Fig. 13 (the values a_{ij} are shown above the lines connecting persons and objects). All persons are assigned as shown, except for person i, who initiates a cooperative auction iteration with $\epsilon = 0$. The starting object prices are $p = (0,\dots,0)$, and satisfy CS together with the partial assignment shown.

Let us apply the cooperative auction algorithm with expanding coalitions and $\epsilon < 0.5$. The starting coalition is $C(i) = \{i, i_1, i_2\}$ and the price rise of the set of objects $\mathcal{O}(i) = \{j_1, j_2\}$ is $r = 0.5$, which brings object j_3 into the 0-zone of person i_2. A new iteration is started by person i, with coalition $C(i) = \{i, i_1, i_2, i_3\}$ and price rise of the set of objects $\mathcal{O}(i) = \{j_1, j_2, j_3\}$ equal to $r = 0.5$, which brings object j_4 into the 0-zone of person i_3. This coalition expansion process continues for $n-3$ iterations, up to when person i_{n-1}

[7] For an illustration of why price wars involve at least two objects, consider the 3×3 problem of Fig. 4. If there were only one valuable object (value C for all persons) and the other two objects were valueless, the type of price war illustrated in the figure would not occur. For an illustration of how a price war can be generated subsequent to aggressive auction iterations, consider the same example with a fourth person added (with identical values as the other three persons) and a fourth object added offering value -1 for all four persons.

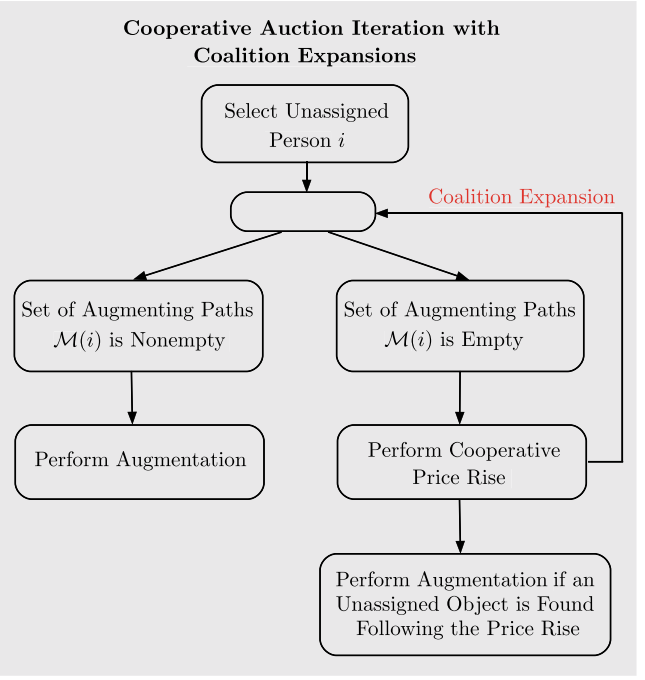


Fig. 11. Block diagram of a cooperative auction iteration with coalition expansions. The iteration always terminates with an augmentation, possibly following multiple coalition expansions.

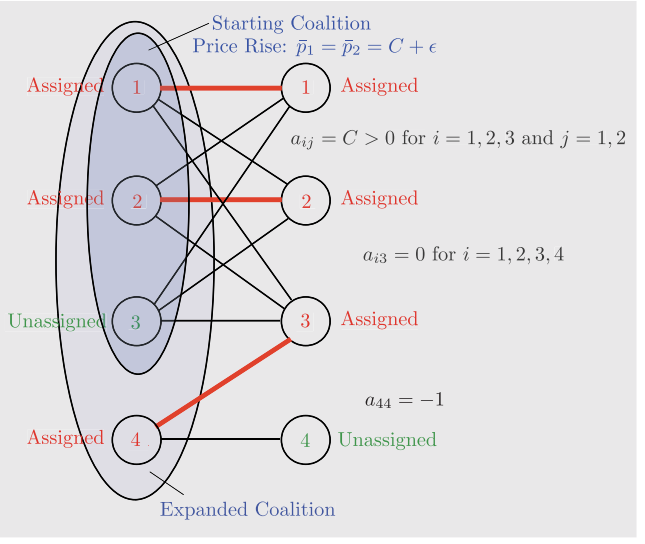


Fig. 12. Illustration of the variant of the cooperative auction iteration that involves an expanding coalition. We consider a 4×4 version of the 3×3 problem of Figs. 2 and 4, as shown above. The initial prices are $p = (0, 0, 0, 0)$ and the initial partial assignment is $\{(1, 1), (2, 2), (4, 3)\}$. We assume that $\epsilon < 1/n = 1/4$ (to guarantee that the final assignment is optimal). The cooperative auction iteration starts with the unassigned person 3, and constructs the coalition of persons 1, 2, and 3, similar to Fig. 8. The prices of objects 1 and 2 rise to $C + \epsilon$, thus bringing the assigned object 3 into the ϵ-zone of the coalition partners 1, 2, and 3. In the expanding coalition variant of the cooperative iteration, the search for coalition partners continues, adding person 4 to the coalition. A new price rise of objects 1, 2, and 3 (by $1 + \epsilon$ units) is then performed. This brings object 4 into the ϵ-zone of person 4 and allows the augmentation $(3, 4, 4)$, and termination with the assignment $\{(1, 1), (2, 2), (3, 3), (4, 4)\}$ and prices $\bar{p} = (C + 1 + 2\epsilon, C + 1 + 2\epsilon, 1 + \epsilon, 0)$. It can be seen that the final assignment and prices satisfy ϵ-CS.

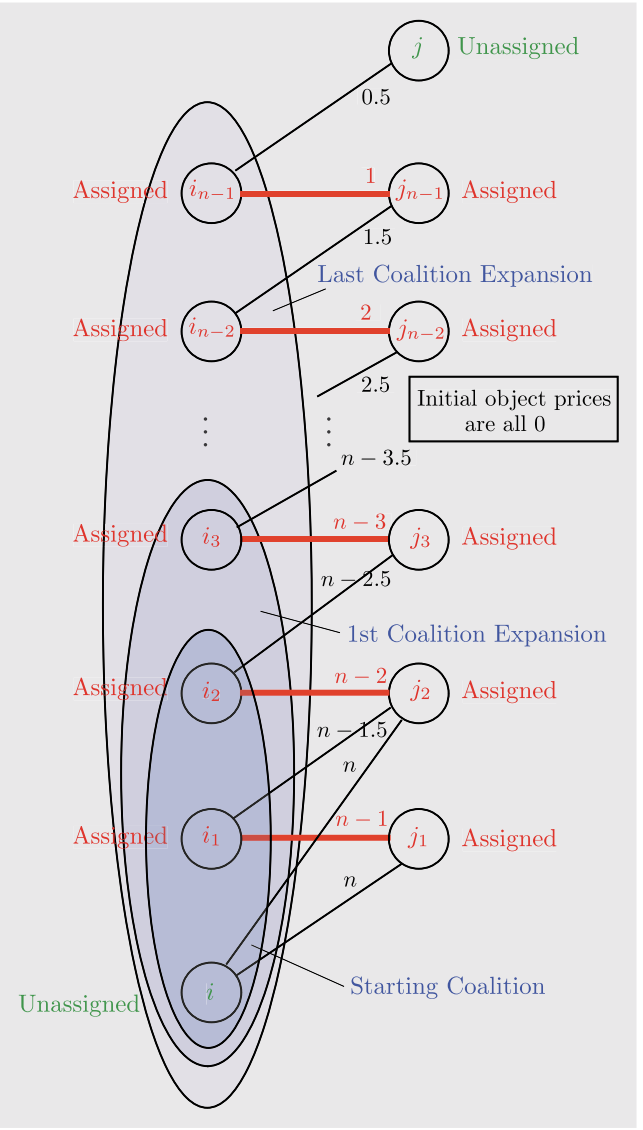


Fig. 13. Illustration of the multiple coalition expansions algorithm with $\epsilon = 0$; cf. Example 3.3. The values a_{ij} are shown above the lines connecting persons and objects, and the initial prices are all 0. The algorithm with multiple coalition expansions requires a single iteration (with multiple coalition expansions) and $O(n)$ computation to assign person i, while the algorithm without coalition expansions (cf. Section 3) requires n iterations and $O(n^2)$ computation.

is included in the coalition, the prices of objects $j_1, \ldots, j_{n-1}$ rise by 0.5, which brings object j into the 0-zone of person i_{n-1}, with an augmentation ensuing along the augmenting path $(i, i_2, i_3, \ldots, i_{n-1}, j)$. The assignment thus obtained is complete and optimal.

This process requires $n-3$ iterations, and $O(n^2)$ computation (because each of the $n-3$ coalitions is rebuilt from scratch). If it is carried out with the expanding coalition variant, it requires a single iteration with $n-3$ coalition expansions, and $O(n)$ computation.

Suppose now that we use $\epsilon \geq 0.5$. Then every object is contained in the ϵ-zone of some person, the starting coalition $C(i)$ is the entire person set $\{i, i_1, \ldots, i_{n-1}\}$, and the algorithm terminates in one iteration, without any coalition expansion. There are two possible augmentations from i to j:

$$(i, i_1, i_2, i_3, \ldots, i_{n-1}, j) \qquad \text{and} \qquad (i, i_2, i_3, \ldots, i_{n-1}, j),$$

and two corresponding complete assignments. The first of these is suboptimal while the second is optimal. The solution generated depends on the order in which persons i_1 and i_2 enter $C(i)$. This illustrates how the number of coalition expansions may be reduced with larger values of ϵ.

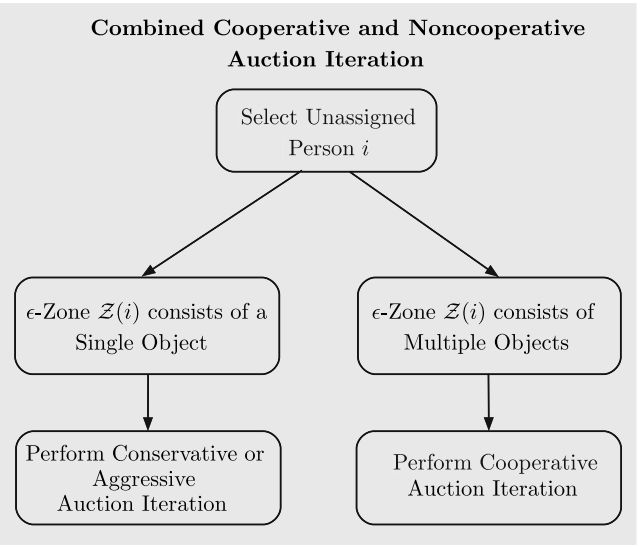


Fig. 14. Block diagram of a combined cooperative and noncooperative auction iteration.

3.2. Combinations with noncooperative auction algorithms

We will now explore the possibility of combining the noncooperative auction algorithms (both conservative and aggressive) with the cooperative algorithm. In particular, we are given a set of object prices $p = (p_1, \ldots, p_n)$ and a partial assignment $\mathcal{A}$ satisfying ϵ-CS. The iteration starts with an unassigned person i and tries to generate the set of coalition partners of i. In the process it will perform an aggressive (or conservative) auction iteration if $\mathcal{Z}(i)$, the ϵ-zone of i, contains a single object and $\epsilon > 0$ (or $\epsilon = 0$, respectively), and a cooperative auction iteration otherwise; see Fig. 14. The intuitive idea is that *when $\mathcal{Z}(i)$ consists of a single object, there can be a most one coalition partner of i, so a price war is not possible*. This favors the use of a noncooperative auction iteration.

The iteration just described in summary is stated in detail as follows.

Combined Cooperative and Noncooperative Auction Iteration
Given a set of object prices $p = (p_1, \ldots, p_n)$ and a partial assignment $\mathcal{A}$ satisfying ϵ-CS, select an unassigned person i. If the ϵ-zone $\mathcal{Z}(i)$ contains a single object perform a noncooperative auction iteration (conservative if $\epsilon = 0$ or aggressive if $\epsilon > 0$). Otherwise perform a cooperative auction iteration.

Note that the iteration can optionally be used with or without coalition expansions. In the former case a cooperative auction iteration is simply continued starting from the same person i, up to the point where an augmentation takes place. It should be noted that the cooperative auction iteration with coalition expansions and $\epsilon = 0$ bears similarity to the Hungarian method, which is typically inferior both in theory and in practice to efficiently implemented aggressive auction iterations. Moreover, its theoretical complexity is known to be inferior to the one of the aggressive auction algorithm. On the other hand, combinations of conservative auction and the Hungarian method have worked well in practice, as verified by the computations given in the author's paper [52], and by the experience with the JV code [56]. The combined iteration given above, with or without coalition expansions, is new for $\epsilon > 0$, and has not been adequately tested, but with proper implementation, is expected to work more efficiently than either one of its cooperative and noncooperative components working in isolation.

The evaluation of the performance of the combined aggressive and cooperative auction iteration, with $\epsilon > 0$ and the expanding coalition process, in conjunction with ϵ-scaling, is an issue of great interest, both theoretically and experimentally. A relevant fact here is that two-phase algorithms, which involve aggressive auction ($\epsilon > 0$) in the first phase and a Hungarian-like algorithm ($\epsilon = 0$) in the second phase after most of the objects have been assigned, have been shown to have computational complexity that is superior to either aggressive auction or the Hungarian method in isolation of each other. In particular, Orlin and Ahuja [77] have derived a related

$$O\big(\sqrt{n}\, m \log(nC)\big) \tag{12}$$

worst-case complexity result for a two-phase algorithm of this type, with the threshold for switching between the two phases skillfully chosen (see also Chapter 5, Exercise 4.5 of the book [66], with solution included in the internet-posted version of the book). The

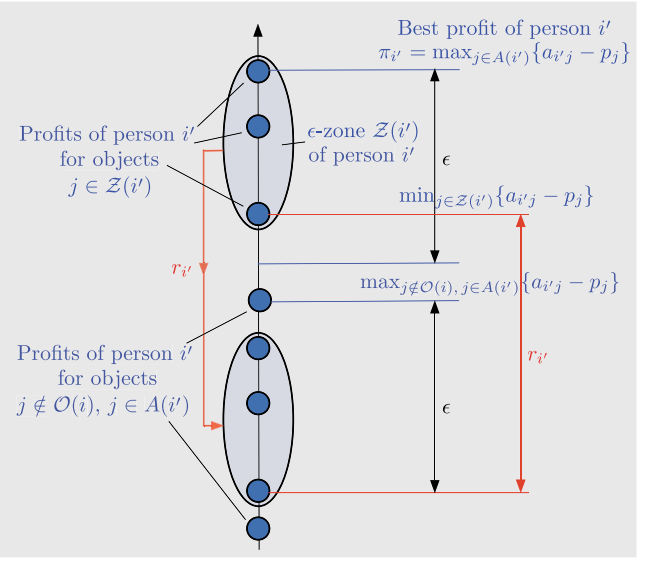


Fig. 15. Illustration of a price rise (and corresponding profit drop) of the objects in the ϵ-zone $\mathcal{Z}(i')$ of a person $i' \in C(i)$. The figure shows the maximum amount $r_{i'}$ by which we can raise the prices of objects j in the ϵ-zone $\mathcal{Z}(i')$, while guaranteeing that all $j \in \mathcal{Z}(i')$ will stay within the ϵ-zone of i' following a cooperative price rise. It is given by $r_{i'} = \epsilon + \min_{j \in \mathcal{Z}(i')}\{a_{i'j} - p_j\} - \max_{j \notin \mathcal{O}(i),\, j \in A(i')}\{a_{i'j} - p_j\}$, cf. Eq. (14). In the above figure the ϵ-zone $\mathcal{Z}(i')$ consists of the three most profitable objects of person i' with profits within the top ellipse. The figure also shows the profits of the objects $j \notin \mathcal{O}(i)$ with $j \in A(i')$ [the profits of any additional objects $j \in \mathcal{O}(i)$ with $j \in A(i')$ but $j \notin \mathcal{Z}(i')$ are not shown]. The figure assumes that the set $\{j \mid j \notin \mathcal{O}(i),\, j \in A(i')\}$ is nonempty; if it is not, we have $r_{i'} = \infty$. After we raise the prices of the objects in $\mathcal{O}(i)$ by $r_{i'}$, the profits of objects in $\mathcal{Z}(i')$ move downward by $r_{i'}$, just within ϵ of the fourth object, which now becomes the most profitable. Price rises by amounts smaller than $r_{i'}$ still keep the three most profitable objects within the ϵ-zone $\mathcal{Z}(i')$, but may not be sufficient to bring the fourth object into $\mathcal{Z}(i')$.

use of $\epsilon > 0$ together with ϵ-scaling, requires fewer coalition expansions, as can be seen from Example 3.3, and seems to be a natural alternative way to deal with a large number of coalition expansions for many problems. Thus it is reasonable to conjecture that a complexity estimate like the one of Eq. (12) can be proved for some version of the combined aggressive and cooperative auction iteration.

3.3. Properties of the cooperative auction algorithm

In this section we will discuss some general properties and implementations of cooperative auction. We first note that if there is no augmenting path starting from i [i.e., $\mathcal{M}(i)$ is empty], the set of objects that are assigned to some coalition partner of i, is the union of the ϵ-zones of the persons in the coalition of i:

$$\mathcal{O}(i) = \cup_{i' \in C(i)} \mathcal{Z}(i'). \tag{13}$$

To see this, note that when $\mathcal{M}(i)$ is empty, all objects in the ϵ-zones of i and his/her coalition partners must be assigned to some coalition partner of i (otherwise an augmentation would be performed).

Let us now provide an explicit formula for the common price rise for the case where $\mathcal{M}(i)$ is empty. For each person $i' \in C(i)$, consider the scalar

$$r_{i'} = \epsilon + \min_{j \in \mathcal{Z}(i')} \{a_{i'j} - p_j\} - \max_{j \notin \mathcal{O}(i),\, j \in A(i')} \{a_{i'j} - p_j\}, \tag{14}$$

(by convention, the maximum above is $-\infty$ if the set over which the maximum is taken is empty). It can be seen from Fig. 15 that $r_{i'}$ is the maximum price rise of the objects in $\mathcal{O}(i)$ that will keep every object in $\mathcal{Z}(i')$, the ϵ-zone of i', within $\mathcal{Z}(i')$ following the price rise. To keep *all* the objects in $\mathcal{O}(i) = \cup_{i' \in C(i)} \mathcal{Z}(i')$ within the ϵ-zone of either i or some coalition partner of i, the common price rise should not exceed any one of the amounts $r_{i'}$, $i' \in C(i)$. Thus the maximum possible common price rise is

$$r = \min_{i' \in C(i)} r_{i'}. \tag{15}$$

Moreover, following the price rise, the union of the ϵ-zones of the persons in $C(i)$ consists of $\mathcal{O}(i)$ and a nonempty set $\overline{\mathcal{O}}(i)$ of additional objects. This is the set of objects that attain the maximum in the maximization of Eq. (14), $\max_{j \notin \mathcal{O}(i),\, j \in A(i'')}\{a_{i''j} - p_j\}$, while i'' attains the minimum in Eq. (15). Thus the set of objects $\overline{\mathcal{O}}(i)$ obtained at the end of the iteration is nonempty and r is finite (otherwise the existence of a complete assignment assumption would be violated). Note that if any of the objects within $\overline{\mathcal{O}}(i)$, say object j, is unassigned, we can perform an augmentation that starts at i and ends at j, and increase the price of j by the maximum amount that will not violate ϵ-CS. This can be done efficiently, and it is generally recommended, as it increases the

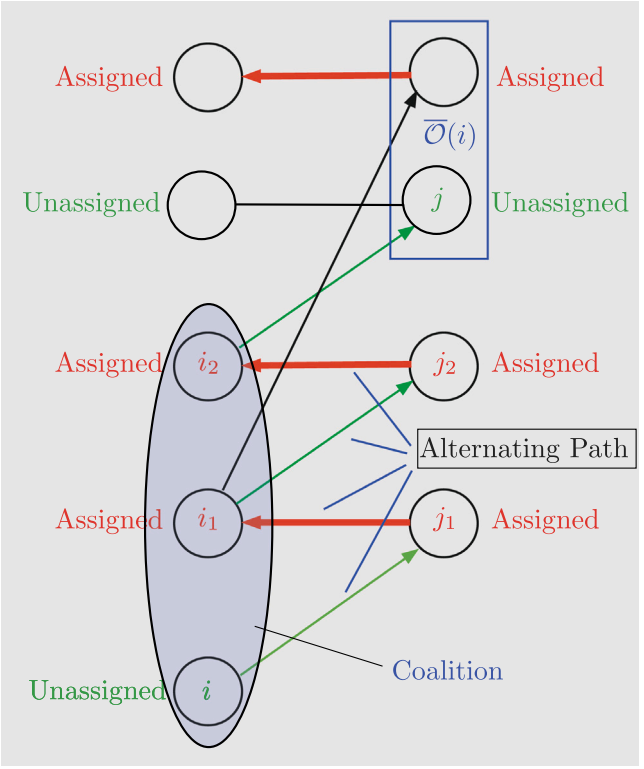


Fig. 16. Illustration of the cooperative auction iteration, where after raising the prices, there is an object in the set $\overline{\mathcal{O}}(i)$ that is unassigned. Then, we may optionally perform an augmentation, shown in green, along a corresponding augmenting path [(i, i_1, i_2, j) in the figure]. In particular, persons i, i_1, and i_2 get assigned to j_1, j_2, and j, respectively. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

cardinality of the assignment by one, but for simplicity, we have not stated this explicitly. Alternatively, we may suitably modify the cooperative iteration description, so an augmentation is automatically performed, if possible, following a price rise. Fig. 16 provides an illustration.

We summarize the principal conclusions from the preceding discussion in the following proposition.

Proposition 3.1. *Consider the cooperative auction iteration under the assumption that there is no augmenting path starting from i, i.e., $\mathcal{M}(i)$ is empty. Consider also $\mathcal{O}(i)$, the set of objects that are assigned to some coalition partner of i. Then following a price rise:*

(a) $\mathcal{O}(i)$ is equal to the union of the ϵ-zones of all persons in $C(i)$.

(b) The prices of the objects in $\mathcal{O}(i)$ are raised by the common increment

$$r = \min_{i' \in C(i)} r_{i'},$$

where $r_{i'}$ is given by Eq. (14), and we have $r > \epsilon$.

(c) Following the price rise, the union of the ϵ-zones of the persons in $C(i)$ consists of $\mathcal{O}(i)$ and a nonempty set $\overline{\mathcal{O}}(i)$ of additional objects.

(d) If any of the objects within $\overline{\mathcal{O}}(i)$, say object j, is unassigned, an augmentation that starts at i and ends at j can be performed. Moreover, the prices and assignment obtained following this augmentation satisfy ϵ-CS.

The computational complexity of the algorithm is not expected to be better than the one of the aggressive auction algorithm [$O\big(nm \log(nC)\big)$, where m is the number of arcs of the bipartite graph representing the assignment problem and C is the range of values, given by Eq. (10)]. However, depending on the implementation and the type of problem addressed, it appears that the cooperative auction algorithm, as given in this section, can outperform the aggressive auction algorithm, particularly in situations where price wars are likely.

3.4. Common price increment computation

We will now focus on the most complicated part of a cooperative auction iteration, namely the computation of the common price rise increment r of Eqs. (14)–(15), and the new set of objects $\overline{\mathcal{O}}(i)$ that are subsequently brought into the coalition of i, when there is no augmenting path starting from i. We will first describe one possible implementation that can be interpreted graphically,

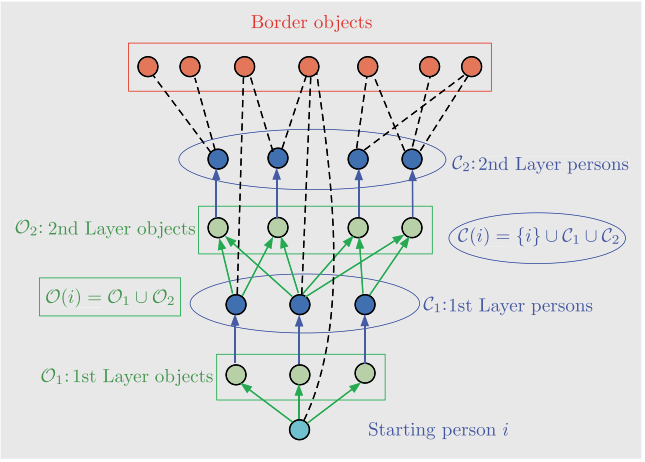


Fig. 17. Illustration of a search tree to construct the coalition of i, $\mathcal{C}(i) = \{i\} \cup \mathcal{C}_1 \cup \mathcal{C}_2$, assuming no augmentation occurs during the cooperative auction iteration (the figure assumes two object layers $\mathcal{O}_1, \mathcal{O}_2$, and two person layers $\mathcal{C}_1, \mathcal{C}_2$). The objects in $\mathcal{O}_1$ are the ones in the ϵ-zone $\mathcal{Z}(i)$. The objects in $\mathcal{O}_2$ are the ones that do not belong to $\mathcal{O}_1$ but belong to the ϵ-zone $\mathcal{Z}(i')$ of some person of $\mathcal{C}_1$. The set $\mathcal{B}$ of border objects consists of all objects j that do not belong to $\mathcal{O}(i) = \mathcal{O}_1 \cup \mathcal{O}_2$, but can be matched with some person $i' \in \mathcal{C}(i)$, i.e., $\mathcal{B} = \big\{j \notin \mathcal{O}(i) \mid j \in A(i') \text{ for some } i' \in \mathcal{C}(i)\big\}$. Green arrows indicate pairs (i', j') such that $i' \in \mathcal{C}(i)$ and $j' \in \mathcal{Z}(i')$. Broken lines indicate pairs (i', j') such that $i' \in \mathcal{C}(i)$, $j' \in A(i')$ but $j' \notin \mathcal{O}(i)$. The paths from i to the (blue) nodes in $\mathcal{C}_1 \cup \mathcal{C}_2$ are the shortest alternating paths. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

and we will subsequently provide a more general implementation in pseudocode. We assume that we are given a set of object prices $p = (p_1, \ldots, p_n)$ and a partial assignment $\mathcal{A}$ satisfying ϵ-CS, together with an unassigned person i to start the iteration. We also assume that the ϵ-zone $\mathcal{Z}(i)$ contains multiple objects all of which are assigned, and that no augmenting path starting from i exists.

In particular, we will use the layered graph shown in Fig. 17 to illustrate the computation of:

(a) The set of coalition persons $\mathcal{C}(i)$.
(b) The set of coalition objects $\mathcal{O}(i)$, i.e., the set of objects assigned to the persons in $\mathcal{C}(i)$.
(c) The common price rise r of the objects in $\mathcal{O}(i)$.

In this computation we break down the sets $\mathcal{O}(i)$ and $\mathcal{C}(i)$ into layers of disjoint subsets $\mathcal{O}_1, \mathcal{C}_1, \ldots, \mathcal{O}_k, \mathcal{C}_k$, where for some positive integer $k < n$, and for $m = 1, \ldots, k$:

The mth person layer $\mathcal{C}_m$ is the set of persons i' such that every alternating path that starts at i and ends at i' contains at least m persons other than i.
The mth object layer $\mathcal{O}_m$ is the set of objects that are assigned to the persons in $\mathcal{C}_m$.

The layers $\mathcal{O}_m$ and $\mathcal{C}_m$ are computed successively, and can be visualized in terms of the tree of alternating paths shown in Fig. 17. The details of the computation are as follows:

Layer Construction

(a) We construct $\mathcal{O}_1$, which is the set of objects in the ϵ-zone $\mathcal{Z}(i)$ of person i, and then $\mathcal{C}_1$, which is the set of persons assigned to the objects in $\mathcal{O}_1$.
(b) Given $\mathcal{C}_m$, we construct $\mathcal{O}_{m+1}$ as the set of objects $j \notin \mathcal{O}_1 \cup \cdots \cup \mathcal{O}_m$ that belong to the ϵ-zone of at least one person in $\mathcal{C}_m$; if $\mathcal{O}_{m+1}$ is empty, then we stop (i.e., $m = k$), having computed $\mathcal{O}(i)$ and $\mathcal{C}(i)$ according to

$$\mathcal{O}(i) = \mathcal{O}_1 \cup \cdots \cup \mathcal{O}_k, \qquad \mathcal{C}(i) = \{i\} \cup \mathcal{C}_1 \cup \cdots \cup \mathcal{C}_k.$$

In the process of constructing the layers $\mathcal{O}_1, \mathcal{C}_1, \ldots, \mathcal{O}_k, \mathcal{C}_k$, we obtain the set of *border objects*, denoted $\mathcal{B}$, and consisting of the objects that do not belong to $\mathcal{O}(i)$ but can be matched with a person in the coalition $\mathcal{C}(i)$, i.e.,

$$\mathcal{B} = \big\{j \notin \mathcal{O}(i) \mid j \in A(i') \text{ for some } i' \in \mathcal{C}(i)\big\};$$

see Fig. 17. The border objects are obtained during the process of constructing the sets $\mathcal{O}(i)$ and $\mathcal{C}(i)$ as described earlier.

Simultaneously with the computation of $\mathcal{O}(i)$, $\mathcal{C}(i)$, and $\mathcal{B}$ as described above, we can also compute the cooperative price rise amount of the iteration using Eq. (15):

$$r = \epsilon + \min_{i' \in \mathcal{C}(i)} \left\{ \hat{\pi}_{i'} + \min_{j \in \mathcal{B},\, j \in A(i')} \{p_j - a_{i'j}\} \right\}, \tag{16}$$

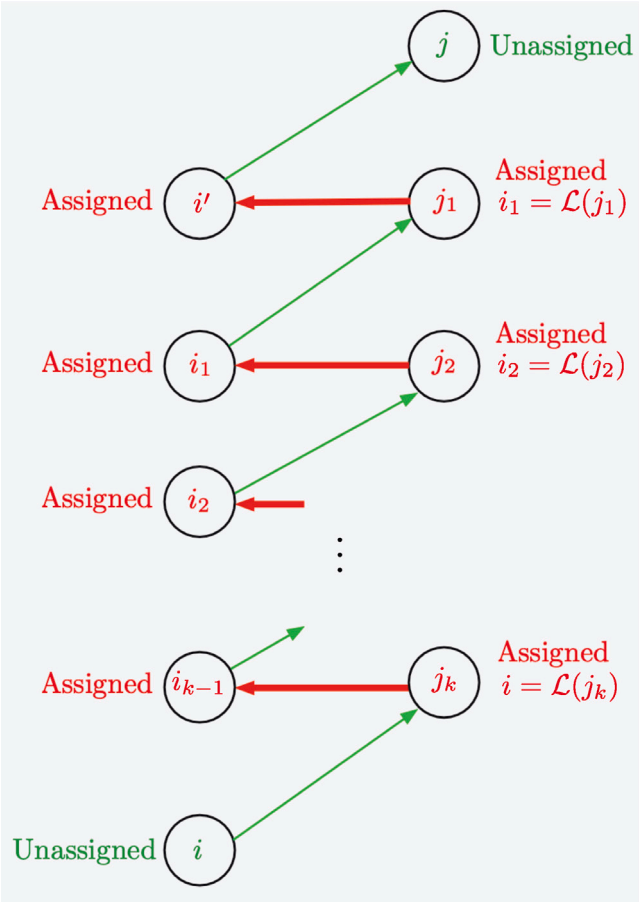


Fig. 18. Illustration of an augmenting path from the unassigned object j to the unassigned person i, which is discovered during an iteration that starts with the unassigned person i.

where $\hat{\pi}_{i'}$ is given by

$$\hat{\pi}_{i'} = \min_{j \in \mathcal{Z}(i')} \{a_{i'j} - p_j\}.$$

Combining the preceding equations with Eq. (15) and interchanging the order of minimizations in Eq. (16), we obtain

$$r = \epsilon + \min_{j \in \mathcal{B}} \min_{i' \in C(i),\, j \in A(i')} \left\{\hat{\pi}_{i'} + p_j - a_{i'j}\right\} = \epsilon + \min_{j \in \mathcal{B}} d_j, \tag{17}$$

where for all $j \in \mathcal{B}$

$$d_j = \begin{cases} \min_{i' \in C(i)} \{\hat{\pi}_{i'} + p_j - a_{i'j}\} & \text{if} j \in A(i') \text{for some} i' \in C(i), \\ \infty & \text{otherwise.} \end{cases}$$

To understand the intuitive meaning of d_j, we first note that $\hat{\pi}_{i'}$ is the profit of person i', assuming i' is awarded the least profitable of the objects in his/her ϵ-zone. Then we can view d_j as a *profit loss* incurred when person i' is reassigned to j from his/her least profitable object within $\mathcal{Z}(i')$. The common price rise r of Eq. (16) can be interpreted as ϵ plus the minimum possible profit loss some person i' is reassigned to some $j \in \mathcal{B}$ from his/her least profitable object in $\mathcal{Z}(i')$. Note also that each reassignment of a person $i' \in C(i)$ to an object in $\mathcal{O}(i)$, in the course of an augmentation, involves a loss or gain in profit of at most ϵ, since the objects assigned to i' before and after the augmentation both belong to the ϵ-zone $\mathcal{Z}(i')$.

Note that $\hat{\pi}_{i'}$ can be computed while we go over the set of associated objects $A(i')$ of person i', to determine whether they can be added to $\mathcal{O}(i)$. Thus the computation of r can be organized progressively: first update the quantity d_j, as new persons i' are added to the coalition $C(i)$, and then at the end of the iteration, after $C(i)$ and $\mathcal{B}$ are obtained, take the minimum over $j \in \mathcal{B}$ of d_j to obtain r; cf. Eq. (17). Also the set of objects $\overline{\mathcal{O}}(i)$ that enter the ϵ-zone of at least one person in the coalition $C(i)$ following the price rise, include the ones that attain the minimum of $d_{j'}$ over $j' \in \mathcal{B}$.

A More General Implementation of the Coalition Construction Process

Let us now provide pseudocode for a more general implementation of the cooperative auction iteration that constructs the sets $C(i)$, $\mathcal{B}$, the scalar r of Eq. (17), and an augmenting path (if one is discovered in the course of the iteration). The code uses two temporary lists of persons C and C'. At the end of the iteration, $C = C(i)$ and $C' = \emptyset$. The code also uses an array $\mathcal{L}$ of labels, with label $\mathcal{L}(j)$ corresponding to object j. The labels $\mathcal{L}(j)$ are initially set to 0, and are updated to record a predecessor person i' of j in a potential augmenting path, i.e., $j \in \mathcal{Z}(i')$. The labels are used to trace backwards an augmenting path from an unassigned object to the unassigned person i that starts the iteration [if one is discovered, in which case we go to the next iteration, without completing the computation of $C(i)$ and $\mathcal{B}$] (see Fig. 18).

Pseudocode to Construct the Sets $C(i)$ and $\mathcal{B}$

Initialization: $C = \emptyset$, $C' = \{i\}$, $\mathcal{B} = \{1, \ldots, n\}$, $d_j = \infty$ and $\mathcal{L}(j) = 0$ for all $j \in \{1, \ldots, n\}$. Until $C' = \emptyset$:

Remove a person i' from C' and add it to C. Let $\hat{\pi}_{i'} = \min_{j \in \mathcal{Z}(i')} \{a_{i'j} - p_j\}$. For all $j \in A(i') \cap \mathcal{B}$:

- If $j \in \mathcal{Z}(i')$ and j is assigned to a person i'', remove j from $\mathcal{B}$, and add i'' to C' if it is not already in C'. Moreover, if $\mathcal{L}(j) = 0$, set $\mathcal{L}(j) = i'$.
- If $j \in \mathcal{Z}(i')$ and j is unassigned, perform an augmentation that starts at i and ends at j, by tracing labels backwards from j to the unassigned person i, along the augmenting path defined as follows:

$$j \to i' \to j_1 = \text{Object Assigned to } i' \to i_1 = \mathcal{L}(j_1)$$

$$\to j_2 = \text{Object Assigned to } i_1 \to i_2 = \mathcal{L}(j_2) \to \cdots \to j_k \to i,$$

  where j_k is an object in the ϵ-zone $\mathcal{Z}(i)$, so $i = \mathcal{L}(j_k)$; see Fig. 18. Go to the next iteration.
- If $j \notin \mathcal{Z}(i')$, set $d_j \leftarrow \min\{d_j, \hat{\pi}_{i'} + p_j - a_{i'j}\}$.

Set $\mathcal{B} \leftarrow \{j \in \mathcal{B} \mid d_j < \infty\}$, $C(i) = C$, $r = \epsilon + \min_{j \in \mathcal{B}} d_j$.

It can be verified that different rules for choosing the person i' to be removed from C' will lead to the same sets $C(i)$ and $\mathcal{B}$, and the same price rise r. On the other hand, one may or may not obtain the layered structure illustrated in Fig. 17, which corresponds to a special rule for choosing i'. This is the rule that removes the persons i' from C' in the same order in which they entered C'. Other rules may also be considered based on a heuristic or more principled rationale in a given problem.

4. Additional auction variants

There are a number of interesting variations of the cooperative auction algorithm, in addition to those we have discussed so far. Most of these variations are aimed at accelerating convergence, mitigating as much as possible the effects of price wars, and enhancing the suitability for parallel computation. Several of these variations have similar theoretical properties. However, their practical performance may be significantly affected by the character of the specific problem that is being solved, such as graph density/sparsity, large/small range of values a_{ij}, and special characteristics of the graph's structure, such as large/small "diameter" (a measure of the average number of hops between two randomly chosen persons).

In what follows in this section we will review a number of algorithmic ideas that form the basis for variants of conservative, aggressive, and cooperative algorithms, and their combinations. The wide spectrum of possibilities suggests a view of an auction algorithmic landscape where there is no universal best choice that works optimally for all problems. This view is supported by extensive computational results in the paper [103], which tested comparatively some (but by no means all) of the algorithmic ideas discussed in the present paper within a broader context of network optimization problems.

Cooperative Auction Iteration With Collective Bidding and Person Reassignments

This variant of the cooperative auction iteration aims to bring it closer to the aggressive auction iteration, at the expense of foregoing the option of expanding coalitions. Consider the cooperative iteration for the case where there is no augmenting path [$\mathcal{M}(i)$ is empty]. Then after the subsequent collective price rise, the union of the ϵ-zones of the persons in $C(i)$ consists of $\mathcal{O}(i)$ and a nonempty set $\overline{\mathcal{O}}(i)$ of additional objects, as we have discussed in Section 3.3. There are now two possibilities:

(a) *There is an unassigned object $\bar{j}$ within the set $\overline{\mathcal{O}}(i)$*. Then as we discussed earlier, an augmenting path is created following the price rise, which starts at i and ends at $\bar{j}$ (cf. Example 3.1 and Fig. 16). This augmentation can be performed immediately, without waiting for the next iteration to discover it.

(b) *All objects in the set $\overline{\mathcal{O}}(i)$ are assigned*. In this case, the cooperative auction algorithm first presented in Section 3 simply goes to the next iteration. However, there is also a possibility to assign person i through a reassignment of the coalition persons and a rearrangement of the corresponding assigned pairs. This is illustrated in Fig. 19, which should be contrasted with Fig. 16.

The corresponding auction algorithm variant is identical to the cooperative auction iteration of Section 3, except for the additional person reassignment process, which is performed following a price rise that does not result in a subsequent augmentation; see Fig. 20. We state this variant formally as follows:

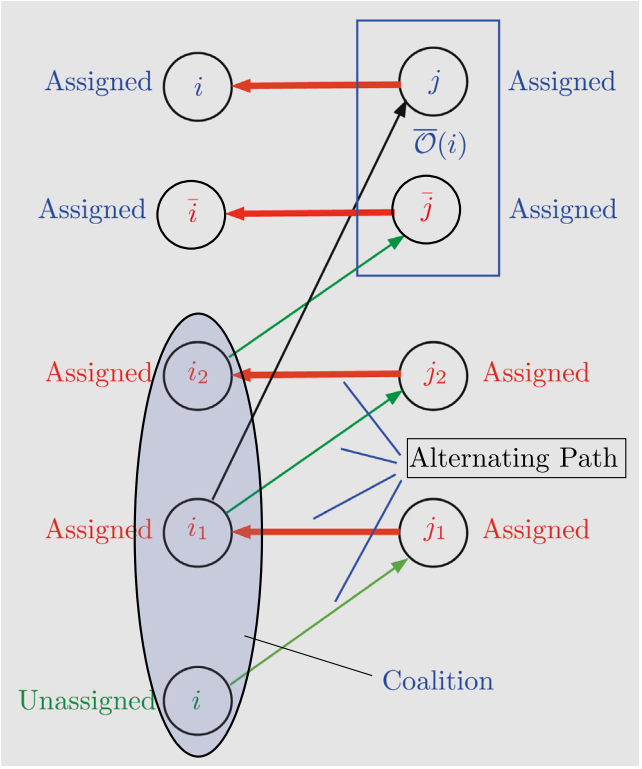


Fig. 19. Illustration of the cooperative auction iteration with person reassignments when the set of augmenting paths $\mathcal{M}(i)$ is empty, and all the objects in the set $\overline{\mathcal{O}}(i)$ are assigned. Then, we choose an object $\bar{j} \in \overline{\mathcal{O}}(i)$ and perform a reassignment of persons to objects, shown in green, along a corresponding alternating path [(i, i_1, i_2) in the figure]. In particular, persons i, i_1, and i_2 get assigned to j_1, j_2, and $\bar{j}$, respectively, while the person $\bar{i}$ that is assigned to $\bar{j}$ under $\mathcal{A}$ becomes unassigned. Note that the object $\bar{j}$ is not unique: any object in $\overline{\mathcal{O}}(i)$ (such as j in the figure) and alternating path corresponding to that object [such as (i, i_1) in the figure] can be used. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Cooperative Auction Iteration With Collective Bidding and Person Reassignments

Given a set of object prices $p = (p_1, \ldots, p_n)$ and a partial assignment $\mathcal{A}$ satisfying ϵ-CS, select an unassigned person i. Let $\mathcal{M}(i)$ be the set of augmenting paths that start with i.

- If $\mathcal{M}(i)$ is nonempty, perform an augmentation along some augmenting path from $\mathcal{M}(i)$, increase the price of the last object in this augmenting path by the maximum amount that will not violate ϵ-CS, and go to the next iteration.
- If $\mathcal{M}(i)$ is empty, let $\mathcal{O}(i)$ denote the set of objects that are assigned to some coalition partner of i. Raise the prices of the objects in $\mathcal{O}(i)$ by the maximum common amount for which the ϵ-zone of every person i' in $C(i)$ is a subset of the ϵ-zone of the same person i' after the price rise.

Let $\overline{\mathcal{O}}(i)$ denote the set of objects $j \notin \mathcal{O}(i)$, which following the price rise, belong to the ϵ-zone of a person in $C(i)$. Select an object $\bar{j} \in \overline{\mathcal{O}}(i)$, with preference given to unassigned objects. Let $(i, i_1, \ldots, i_k)$ be an alternating path such that $\bar{j}$ is in the ϵ-zone of person i_k following the price rise of the coalition objects $\mathcal{O}(i)$. Let also $j_1, \ldots, j_k$ be the objects that are assigned to the persons $i_1, \ldots, i_k$ in the current assignment $\mathcal{A}$. Then change $\mathcal{A}$ by assigning i to j_1, i_m to j_{m+1} for $m = 1, \ldots, k-1$, and i_k to $\bar{j}$; any person assigned to $\bar{j}$ under $\mathcal{A}$ becomes unassigned. Finally, raise the price of $\bar{j}$ by the maximum amount that will not violate ϵ-CS, and go to the next iteration.

The person reassignments in the preceding iteration can be viewed as a collective bid, which aims to acquire a new object for the coalition $C(i)$, at the expense of deassigning a person from outside the coalition. The reassignments bring the iteration closer in spirit to the aggressive auction algorithm. In particular, it can be seen that *if $\mathcal{Z}(i)$ consists of a single object (assigned or unassigned), the preceding iteration behaves identically with the aggressive auction iteration*. On the other hand, we should also note that person reassignments do not allow the use of coalition expansions.

ϵ-Scaling Variations and Auction Initialization

The use of ϵ-scaling may or may not be necessary for the cooperative auction algorithm of Section 3 and its variants. After all, with $\epsilon = 0$ the cooperative algorithm is known to be reliable and to perform well for many problems, particularly those involving a dense assignment graph (this has been established by a number of studies starting with the original paper [52]). On the other hand,

ϵ-scaling may be needed to improve the robustness and the performance of both the aggressive and the cooperative algorithms for the case of a sparse assignment graph.

A critical step in ϵ-scaling is when a complete assignment is obtained with some value of ϵ and then, to run the algorithm with a smaller value $\bar{\epsilon} < \epsilon$, one must discard from the assignment those pairs that do not satisfy $\bar{\epsilon}$-CS. An alternative possibility is to use a variant of the auction algorithm that does not require that the initial price and assignment satisfy $\bar{\epsilon}$-CS. In this variant, we try to execute the cooperative and noncooperative iterations as if $\bar{\epsilon}$-CS were satisfied, and when assigned pairs (i, j) not satisfying $\bar{\epsilon}$-CS are encountered, to discard these pairs from the assignment as needed, while making sure that all newly assigned pairs satisfy $\bar{\epsilon}$-CS. With this somewhat speculative mode of operation, progress can be made towards satisfying $\bar{\epsilon}$-CS as the algorithm is running, with potentially significant computational savings. An additional advantage of this type of scheme is that it can be operated as an "anytime algorithm", i.e., an algorithm that progressively improves on a feasible solution, and returns a complete assignment even if it is interrupted because a computational budget limit or other real-time constraint has been reached.

Variations of ϵ-scaling such as the preceding one can also be helpful when favorable initial conditions prices and assignment pairs are known, which, however, do not satisfy ϵ-CS. For example, good initial conditions may be available by using a trained neural network, which accepts the problem data and provides an approximately optimal (complete or partial) assignment, and corresponding prices, which together may not fully satisfy ϵ-CS for a desired value of ϵ.

In a related context, which is very common in practice, assignment problems are solved repeatedly with small variations in the problem's data (such as small changes in the problem's graph or values). Then there is much to be gained by reusing information in the form of prices and assignment pairs, even if they do not satisfy ϵ-CS. As an example, the author's paper [104] has introduced auction algorithms for path construction and shortest path problems, where the initial conditions need not satisfy ϵ-CS, but are progressively rectified in the course of the algorithm. This is particularly convenient in on-line applications where the problem data changes and maintaining ϵ-CS at all times is difficult (a knowledge graph context of this type is considered by Agarwal, Bertsekas, and Liu [105]). The ideas of the paper [104] (and related ideas from an earlier max-flow paper by the author [106]) can be extended to the algorithms of the present paper for solving assignment problems as well as other network optimization problems.

Adaptive ϵ-Scaling

One possibility to improve the performance of ϵ-scaling schemes is to introduce adaptivity, whereby the value of ϵ is modified in the course of the algorithm, depending on algorithmic progress. In particular, we may start with a small value of ϵ and suitably increase it if some heuristic criterion suggests that a price war is underway (a simple heuristic of this type is implemented in the author's FORTRAN codes noted earlier).

Another possibility is to *use a person-dependent value of* ϵ, so each person has his/her own value that determines the size of his/her ϵ-zone. In particular, if the parameter value ϵ_i is used by person i, we may increase ϵ_i by some factor (up to some upper bound), each time i submits an aggressive auction single-person bid, thereby expanding the ϵ-zone $\mathcal{Z}(i)$. This enhances the cooperative character of iterations that involve repeat bidders, such as the ones participating in a price war. Intuitively, in this form of adaptive ϵ-scaling, *a person i that submits an aggressive bid repeatedly, only to be outbid later by some other person, seeks coalition partners by increasing ϵ_i in order to get through a price war more quickly*.

Reverse Iterations, Similar Persons and Objects, Third Best Profit Test

We mention some additional variants of the auction algorithm, which have been proposed in the literature, and can be adapted to the cooperative framework of this paper and its extensions to other network optimization problems. An important variation involves the use of reverse iterations (see Bertsekas, Castañon, and Tsaknakis [107], and the books [44], Section 4.2, and [45], Section 7.2). In the (forward) auction iterations that we have described so far, persons compete for objects by bidding and raising the prices of objects. In reverse auction iterations, roughly speaking, the *objects compete for persons by essentially offering discounts*.

We can describe reverse auction in two equivalent ways: one where unassigned objects lower their prices as much as possible to attract an unassigned person or to lure a person away from its currently held object 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.

Mathematically, reverse auction is equivalent to forward auction with the roles of persons and objects, and the roles of profits and prices interchanged. On the other hand a typically more effective algorithmic scheme is obtained when forward and reverse auction are combined in an algorithm that switches from forward to reverse auction and back at suitable times. Such a combined algorithm simultaneously maintains a partial assignment, a price vector p, and a profit vector π satisfying the following ϵ-CS condition:

$$\pi_i + p_j \geq a_{ij} - \epsilon, \qquad \text{for all } i \text{ and } j \in A(i), \tag{18}$$

$$\pi_i + p_j = a_{ij}, \qquad \text{for all assigned pairs } (i, j). \tag{19}$$

When forward iterations are used, the prices of objects are increased according to the rules that we have described, while the profits of persons are subsequently reduced to maintain the conditions (18)–(19). When reverse iterations are used, the profits of persons are increased according to rules that can be viewed as "reverse" from the rules we have described, while the prices of objects are subsequently reduced to maintain the conditions (18)–(19).

A simple way to guarantee the validity of such a combined algorithm is to refrain from switching from one type of the auction to the other until the number of assigned person-object pairs increases by at least one. We refer to the books [44], Section 4.2, and [45], Section 7.2, for related analysis and discussion. In practice, combined forward/reverse auction algorithms are affected less by price wars and often work substantially faster than the forward versions. Price wars can still occur in combined forward/reverse algorithms, but they arise through more complex and unlikely problem structures than in forward algorithms. For this reason

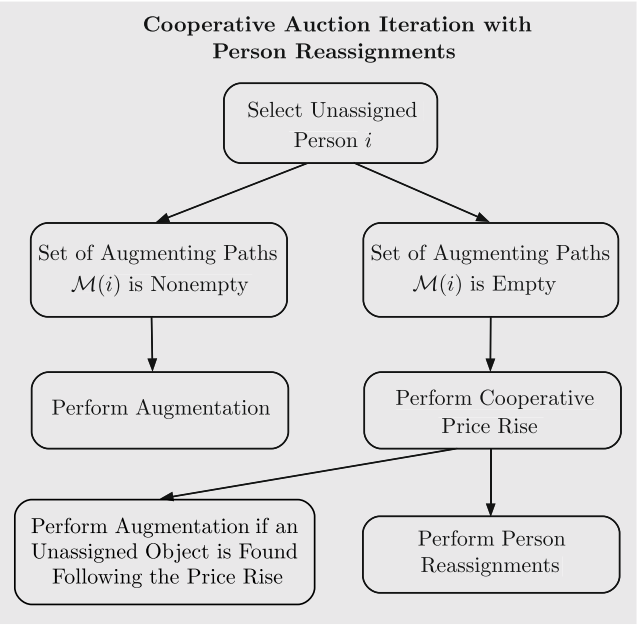


Fig. 20. Block diagram of a cooperative auction iteration with person reassignments. Here, if an unassigned object is not discovered following a cooperative price rise, person i is assigned through a reassignment of the coalition persons and a rearrangement of the corresponding assigned pairs.

combined forward/reverse auction algorithms depend less on ϵ-scaling for good performance than their forward counterparts. In fact, starting with $\epsilon < 1/n$, thus bypassing ϵ-scaling, is sometimes the best choice.

Some variations that are important from both the algorithmic and the theoretical/conceptual point of view deal with problems where there are many "similar" persons and objects [many persons i with identical object sets $A(i)$ and values a_{ij}, $j \in A(i)$]. Problems of this type are particularly susceptible to price wars; see the books [44], Section 4.2, [45], Chapter 7. The paper by Bertsekas and Castañon [108], and the more recent papers by Walsh and Dieci [11,12] propose related auction algorithms in the context of transportation problems, which can be converted into assignment problems with many similar persons and objects. Walsh has also written publicly available auction codes for transportation problems; see https://github.com/jdwalsh03/auction. Alternatively, transportation problems may be viewed as special cases of linear single commodity network problems, and they can be addressed by corresponding natural extensions of auction algorithms.

Finally, we mention another variation of the aggressive auction iteration, which is based on the "third best" profit test (see Exercise 1.7, Section 4.1 of the book [44], or Exercise 7.7 of the book [45]). The motivation here is that frequently in the auction algorithm the two best objects for a given person do not change between two successive bids of that person. The third best test aims to exploit this fact by checking whether the two best objects from the preceding bid continue to be best. If the test is passed, the computation of the profits $a_{ij} - p_j$ of the remaining objects j is unnecessary.

In particular, suppose that at a given aggressive auction iteration that starts with the unassigned person i, we compute the best and second best profits

$$\max_{j \in A(i)} \{a_{ij} - p_j\}, \qquad \max_{j \in A(i), j \neq j_1} \{a_{ij} - p_j\},$$

the corresponding best and second best objects

$$j_1 = \arg \max_{j \in A(i)} \{a_{ij} - p_j\}, \qquad j_2 = \arg \max_{j \in A(i), j \neq j_1} \{a_{ij} - p_j\},$$

and the *third best profit*

$$y_i = \max_{j \in A(i), j \neq j_1, j \neq j_2} \{a_{ij} - p_j\}.$$

Suppose that at a subsequent iteration when person i bids based on an updated price vector $\overline{p}$, we have

$$a_{ij_1} - \overline{p}_{j_1} \geq y_i, \qquad a_{ij_2} - \overline{p}_{j_2} \geq y_i.$$

Because person i's profits can only decrease when passing from the price vector p to the price vector $\overline{p}$, this guarantees that j_1 and j_2 continue to be the two best objects for i (although j_1 may become worse than j_2, based on the updated price vector $\overline{p}$). As a result no further computation is needed to execute the aggressive auction iteration. The third best profit test has proved to be quite effective in practice, requires minimal additional overhead, and has been implemented in the author's FORTRAN codes.

Special Choices of Unassigned Persons

All the algorithms that we have discussed so far, except for the ones involving expanding coalitions, leave open the choice of the unassigned person i that initiates the auction iteration. However, problems with special structure may lend themselves to special/favorable choices of i. For example in assignment problems that have a path construction structure, such as shortest path-type or max-flow-type problems, it may be beneficial to choose unassigned persons in a sequence that corresponds to a candidate solution path or candidate augmenting path; see the author's paper [104] for related auction algorithmic ideas.

In the context of the assignment problem, a special choice of this type corresponds to choosing the person i that starts an auction iteration to be one that has just lost his/her assigned object due to an aggressive bid by another person. We will not go into further details, and instead refer to the papers [103,104,106,109], and the books [44,45] for discussion of such possibilities and the intuition behind them.

Heuristic Criteria for Switching to Cooperative Auction

An issue that arises in combinations of conservative/aggressive and cooperative auction is how to control the switch from one type of auction to another. One possibility is to forgo the aggressive iteration and do a cooperative iteration instead, if some heuristic criterion suggests that a price war is underway; for example, a relatively large number of aggressive iterations that do not produce an augmentation. This is similar to what is done in two-phase auction algorithms with $\epsilon = 0$, which start as single-person/conservative auction and switch to a cooperative auction if price wars persist, e.g., the algorithms of [52,56].

Dealing with Infeasibility

Let us consider the case of an infeasible problem, where there does not exist a complete assignment. In this case, the auction algorithm cannot possibly terminate. It will keep on increasing the prices of some objects by increments of at least ϵ. Furthermore, some persons will be submitting bids infinitely often, and the corresponding profits will be decreasing toward $-\infty$. Methods to detect infeasibility of a given problem have been developed and have been discussed in several of the author's works; see for example [44,45,68]. These methods can be easily incorporated into the algorithmic framework of this paper.

A simple method to deal with infeasibility is to convert the problem to an equivalent feasible problem by adding a set of artificial person-object pairs to the original set of pairs. The values of these pairs should be very small, so that none of them participates in an optimal assignment unless the problem is infeasible. We refer to Section 3.3 of the tutorial paper [68] for further discussion. An alternative possibility is to first check for feasibility of the problem (before attempting to solve it) by using a low complexity bipartite matching algorithm for infeasibility detection.

Finally, let us note that if the expanding coalitions variant is used, the detection of infeasibility is simple: the problem is infeasible if and only if in the course of some cooperative iteration (with coalition expansion) we encounter an empty set of border nodes; this can only happen if there is no complete assignment cf. Proposition 3.1(c).

5. Concluding remarks

We have introduced a new cooperative auction iteration, and variations thereof, for symmetric linear assignment problems, which may use a positive value of ϵ, and can resolve competitive impasses and price wars without requiring the use of ϵ-scaling (although it can be used in conjunction with ϵ-scaling). The iteration is recommended when the ϵ-zone of the starting unassigned person consists of multiple objects, all of which are assigned, an indication of the possibility of a price war; otherwise the classical aggressive form of the auction iteration is typically preferable. The variant of the cooperative auction iteration that involves person reassignments actually coincides with the aggressive auction iteration when the ϵ-zone of the starting person consists of a single object.

The auction iterations described in this paper admit extensions to other classical network optimization problems such as asymmetric assignment, multiassignment, shortest path, k-shortest path, max-flow, and transportation problems. All of these problems can in turn be viewed as special cases of the general single commodity linear network flow problem, which is commonly referred to as the *minimum cost flow problem* (MCNF for short) in the literature.

We plan to discuss extensions of the cooperative auction algorithm and its variants to other network flow problems in future publications. However, it is worth mentioning here some connections between the assignment algorithms of the present paper and algorithms for the MCNF problem, which point the way to future work:

(a) Conservative auction, when generalized to the MCNF problem, becomes the single node relaxation method described in Section 6.3 of the book [45].

(b) Aggressive auction, when generalized to the MCNF problem, becomes the ϵ-relaxation method first proposed by the author in the paper [110], and described and analyzed in detail in the books [66] (Sections 5.3, 5.4), [44] (Section 4.5), and [45] (Section 7.4). This method is also closely related to preflow-push methods, as noted earlier.

(c) The cooperative auction algorithm with $\epsilon = 0$ and no coalition expansions, when generalized to the MCNF problem, becomes the relaxation method of Bertsekas [53], and Bertsekas and Tseng [54]; see also the books [44] (Section 3.3) and [45] (Section 6.3).

(d) The cooperative auction algorithm with $\epsilon = 0$ and coalition expansions, when generalized to the MCNF problem, becomes the classical primal–dual (sequential shortest path) method; see also the books [44] (Section 3.2) and [45] (Section 6.2).

(e) The variant of the cooperative auction algorithm that was first presented in Section 4 (person reassignments along an alternating path), when generalized to the MCNF problem with $\epsilon = 0$, becomes a variant of the relaxation method described by the author in the paper [103] under the name "early flow augmentations".

(f) An auction algorithm for the max-flow problem, given by the author in the paper [106], combines several of the variations of aggressive and cooperative auction algorithms that we have discussed. Of course, the max-flow problem has special structure (such as zero arc costs and hence no need for ϵ-scaling), which can be exploited when specializing the algorithms of the present paper to its context.

(g) Cooperative auction with $\epsilon > 0$, and its variants with and without coalition expansions and person reassignments, are new algorithms, which generalize without much difficulty to the MCNF problem and its special cases noted earlier. Early ideas in this regard can be found in the paper by Bertsekas and Castañon [111], and the book [45], Section 9.6.

Another form of extension to a MCNF problem that involves a convex (rather than linear) separable cost function, is also possible. It can be based on related problem transformation ideas (see the papers by Bertsekas, Polymenakos, and Tseng [64,65], and the textbook [45], Chapter 9).

A basic mechanism for extension of auction algorithms to MCNF problems and special cases thereof is to first convert such problems to assignment problems, by using well known transformations, then apply one of the algorithms of the present paper, and then streamline the computations for efficiency. However, as a practical matter one should not try to literally convert one of the assignment algorithms of the present paper to a new problem structure. Instead one should aim to combine and adapt the principal algorithmic ideas presented in this paper, in sensible ways that experimentally can be shown to work well for the given type of problem. These ideas are conservative, aggressive, and cooperative price rises and augmentations, under the umbrella of the mathematically fundamental approximation framework of ϵ-CS, and the intuitive framework of auction-based economic competition.

Let us also mention extensions of the cooperative auction iteration (possibly in combination with aggressive auction iterations) that may involve multiple unassigned persons. These persons may submit bids in parallel or distributed, possibly asynchronous, fashion. Extensions of this type are not considered in the present paper. We refer to the book [66], Sections 5.3 and 6.5, for related discussions of distributed asynchronous aggressive auction algorithms, and also the papers by Bertsekas and Castañon [112,113] for distributed asynchronous implementations of the Hungarian method and primal–dual methods.

We note that beyond their use in addressing the MCNF problem, our algorithmic ideas lend themselves well for incorporation in heuristics for assignment-like problems, which are more difficult than the linear assignment problem that we have considered in this paper. Such problems include multi-dimensional assignment, combinatorial auctions, dynamic task allocation, and multiagent/multi-robot problems. A noteworthy approach is to use an auction algorithm as a base heuristic for a rollout algorithm; see the books [45] (Section 10.5), [80] (Section 3.4.2), and [114] (Chapter 2).

Finally, let us mention an interesting connection with reinforcement learning. One of the important favorable characteristics of auction algorithms is that the final prices obtained from solution of a given assignment problem can be used as initial prices for applying the algorithms to other problems, which are structurally similar. This suggests that one may try to "learn" favorable initial prices from data, which encode this knowledge into a neural network that can supply on demand good initial prices for a given problem and (possibly) an associated favorable partial assignment. Work on machine learning and neural network approaches towards assignment problems is at a very early stage at present; see e.g., Lee et al. [115], Emami et al. [90], and Aironi, Cornell, and Squartini [19]. It is reasonable to expect that auction algorithms and their intuitive economic competition-like mechanism lend themselves well to this line of research.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

No data was used for the research described in the article.

References

[1] Brenier Y, Frisch U, Henon M, Loeper G, Matarrese S, Mohayaee R, et al. Reconstruction of the early universe as a convex optimization problem. Mon Not R Astron Soc 2003;346:501–24.
[2] Frisch U, Sobolevskii A. Application of optimal transport theory to reconstruction of the early universe. J Math Sci 2006;133:1539–42.
[3] Lavaux G. Lagrangian reconstruction of cosmic velocity fields. Physica D 2008;237:2139–44.
[4] Villani C. Optimal transport: Old and new. Berlin: Springer; 2009.
[5] Villani C. Topics in optimal transportation. American Mathematical Society; 2021.
[6] Santambrogio F. Optimal transport for applied mathematicians. Springer Intern. Publ; 2015.
[7] Galichon A. Optimal transport methods in economics. Princeton University Press; 2016.
[8] Metivier L, Brossier R, Merigot Q, Oudet E. Graph space optimal transport for FWI: Auction algorithm, application to the 2d valhall case study. In: 81st EAGE conference and exhibition. European Association of Geoscientists and Engineers; 2019.
[9] Schmitzer B. A sparse multiscale algorithm for dense optimal transport. J Math Imaging Vision 2016;56:238–59.
[10] Schmitzer B. Stabilized sparse scaling algorithms for entropy regularized transport problems. SIAM J Sci Comput 2019;41:A1443–81.
[11] Walsh JD, Dieci L. General auction method for real-valued optimal transport. 2017, arXiv preprint arXiv:1705.06379.
[12] Walsh JD, Dieci L. A real-valued auction algorithm for optimal transport. Stat Anal Data Min ASA Data Sci J 2019;12(6):514–33.
[13] Peyre G, Cuturi M. Computational optimal transport: With applications to data science. Found Trends Mach Learn 2019;11:355–607.
[14] Levy B, Mohayaee R, Hausegger Svon. A fast semidiscrete optimal transport algorithm for a unique reconstruction of the early universe. Mon Not R Astron Soc 2021;506:1165–85.

[15] Merigot Q, Thibert B. Optimal transport: Discretization and algorithms. In: Handbook of numerical analysis, vol. 22, Elsevier; 2021, p. 133–212.
[16] Kollias G, Sathe M, Schenk O, Grama A. Fast parallel algorithms for graph similarity and matching. J Parallel Distrib Comput 2014;74:2400–10.
[17] Erciyes K. Distributed and sequential algorithms for bioinformatics. Springer; 2015.
[18] Zhou J, Cui G, Hu S, Zhang Z, Yang C, Liu Z, et al. Graph neural networks: A review of methods and applications. AI Open 2020;1:57–81.
[19] Aironi C, Cornell S, Squartini S. Tackling the linear sum assignment problem with graph neural networks. In: International conference on applied intelligence and informatics. Springer; 2022, p. 90–101.
[20] Nurlanov Z, Schmidt FR, Bernard F. Universe points representation learning for partial multi-graph matching. In: Proc. of the AAAI conference on artificial intelligence, vol. 37. 2023, p. 1984–92.
[21] Parkes DC, Ungar LH. Iterative combinatorial auctions: Theory and practice, vol. 7481, Aaai/iaai; 2000, p. 53.
[22] Vries SDe, Vohra RV. Combinatorial auctions: A survey. INFORMS J Comput 2003;15:284–309.
[23] Kosowsky JJ, Yuille AL. The invisible hand algorithm: Solving the assignment problem with statistical physics. Neural Netw 1994;7:477–90.
[24] Jacobs M, Merkurjev E, Esedoglu S. Auction dynamics: A volume constrained MBO scheme. J Comput Phys 2018;354:288–310.
[25] Bertozzi AL, Merkurjev E. Graph-based optimization approaches for machine learning, uncertainty quantification and networks. In: Handbook of numerical analysis, vol. 20. 2019, p. 503–31.
[26] Merkurjev E. A fast graph-based data classification method with applications to 3D sensory data in the form of point clouds. Pattern Recognit Lett 2020;136:154–60.
[27] Bayati M, Prabhakar B, Shah D, Sharma M. Iterative scheduling algorithms. In: IEEE INFOCOm 2007-26th IEEE international conf. on computer communications. 2007, p. 445–53.
[28] Bayati M, Shah D, Sharma M. Max-product for maximum weight matching: Convergence, correctness, and LP duality. IEEE Trans Inform Theory 2008;54:1241–51.
[29] Choi HL, Brunet L, How JP. Consensus-based decentralized auctions for robust task allocation. IEEE Trans Robot 2009;25:912–26.
[30] Liu L, Shell DA. Optimal market-based multi-robot task allocation via strategic pricing. Robotics: Sci Syst 2013;9:33–40.
[31] Luo L, Chakraborty N, Sycara K. Provably-good distributed algorithm for constrained multi-robot task assignment for grouped tasks. IEEE Trans Robot 2014;31:19–30.
[32] Morgan D, Subramanian GP, Chung SJ, Hadaegh FY. Swarm assignment and trajectory optimization using variable-swarm, distributed auction assignment and sequential convex programming. Int J Robot Res 2016;35:1261–85.
[33] Tang J, Zhu K, Guo H, Gong C, Liao C, Zhang S. Using auction-based task allocation scheme for simulation optimization of search and rescue in disaster relief. Simul Model Pract Theory 2018;82:132–46.
[34] Duan X, Liu H, Tang H, Cai Q, Zhang F, Han X. A novel hybrid auction algorithm for multi-UAVs dynamic task assignment. IEEE Access 2019;8:86207–22.
[35] Huang Y, Zhang Y, Xiao H. Multi-robot system task allocation mechanism for smart factory. In: 2019 IEEE 8th joint international information technology and artificial intelligence conference. 2019, p. 587–91.
[36] Lujak M, Matezovic M. On efficiency in dynamic multi-robot task allocation. In: AIRO proceedings. 2020, p. 49–53.
[37] Lujak M, Giordani S, Omicini A, Ossowski S. Decentralizing coordination in open vehicle fleets for scalable and dynamic task allocation. Complexity 2020;1–21.
[38] Otte M, Kuhlman MJ, Sofge D. Auctions for multi-robot task allocation in communication limited environments. Auton Robots 2020;44:547–84.
[39] Aziz H, Pal A, Pourmiri A, Ramezani F, Sims B. Task allocation using a team of robots. Curr Robot Rep 2022;3:227–38.
[40] Wang C, Mei D, Wang Y, Yu X, Sun W, Wang D, et al. Task allocation for multi-AUV system: A review. Ocean Eng 2022;266(12911).
[41] Garces D, Bhattacharya S, Gil S, Bertsekas DP. Multiagent reinforcement learning for autonomous routing and pickup problem with adaptation to variable demand. In: 2023 IEEE international conference on robotics and automation. 2023, p. 3524–31.
[42] Li H, Zhu H, Xu D, Lin X, Jiao G, Song Y, et al. Dynamic task allocation based on auction in robotic mobile fulfilment system. J Ind Manag Optim 2023;19.
[43] Wang Y, Li H, Yao Y. An adaptive distributed auction algorithm and its application to multi-AUV task assignment. Sci China Technol Sci 2023;1–10.
[44] Bertsekas DP. Linear network optimization. Cambridge, MA: MIT Press; 1991.
[45] Bertsekas DP. Network optimization: Continuous and discrete models. Belmont, MA: Athena Scientific; 1998, (can be downloaded from the author's website).
[46] Bertsimas D, Tsitsiklis JN. Introduction to linear optimization. Belmont, MA: Athena Scientific; 1997.
[47] Burkard R, Amico MDell', Martello S. Assignment problems. Society for Industrial and Applied Mathematics; 2012.
[48] Ahuja RK, Magnanti TL, Orlin JB. Network flows. dspace.mit.edu; 1988.
[49] Ahuja RK, Magnanti TL, Orlin JB. Network flows. In: Nemhauser GL, et al., editors. Handbooks in operations research and management science, vol. 1, Optimization. Amsterdam: North-Holland; 1989, p. 211–369.
[50] Burkard RE, Cela E. Linear assignment problems and extensions. In: Handbook of combinatorial optimization, supplement vol. A. Boston: Springer; 1999, p. 75–149.
[51] Kuhn HW. The hungarian method for the assignment problem. Nav Res Logist Q 1955;2:83–97.
[52] Bertsekas DP. A new algorithm for the assignment problem. Math Program 1981;21:152–71.
[53] Bertsekas DP. A unified framework for minimum cost network flow problems. Math Program 1985;32:125–45.
[54] Bertsekas DP, Tseng P. Relaxation methods for minimum cost ordinary and generalized network flow problems. Oper Res 1988;36:93–114.
[55] Bertsekas DP. A distributed algorithm for the assignment problem. Lab. for information and decision systems report, MIT; 1979.
[56] Jonker R, Volgenant T. A shortest augmenting path algorithm for dense and sparse linear assignment problems. Computing 1987;38.
[57] Bertsekas DP, Eckstein J. Dual coordinate step methods for linear network flow problems. Math Program B 1988;42:203–43.
[58] Castañon DA. Reverse auction algorithms for assignment problems. In: Johnson DS, McGeoch CC, editors. Algorithms for network flows and matching. Providence, RI: American Math. Soc.; 1993, p. 407–29.
[59] Zaki HA. A comparison of two algorithms for the assignment problem. Comput Optim Appl 1995;4:23–45.
[60] Malkoff DB. Evaluation of the Jonker-Volgenant-Castañon (JVC) assignment algorithm for track association. In: Proc. SPIE 3068, signal processing, sensor fusion, and target recognition. 1997, http://dx.doi.org/10.1117/12.280801.
[61] Bertsekas DP. Mathematical equivalence of the auction algorithm for assignment and the epsilon-relaxation (preflow-push) method for min cost flow. In: Hager WW, Hearn DW, Pardalos PM, editors. Large scale optimization. Boston, MA: Springer; 1993.
[62] Naparstek O, Leshem A. Expected time complexity of the auction algorithm and the push relabel algorithm for maximum bipartite matching on random graphs. Random Struct Algorithms 2016;48:384–95.
[63] Alfaro CA, Perez SL, Valencia CE, Vargas MC. The assignment problem revisited. Optim Lett 2022;16:1531–48.
[64] Bertsekas DP, Polymenakos LC, Tseng P. An epsilon-relaxation method for convex network optimization problems. SIAM J Optim 1997;7:853–70.
[65] Bertsekas DP, Polymenakos LC, Tseng P. Epsilon-relaxation and auction methods for separable convex cost network flow problems. In: Pardalos PM, Hearn DW, Hager WW, editors. Network optimization. Lecture notes in economics and mathematical systems, N.Y: Springer-Verlag; 1998, p. 103–26.
[66] Bertsekas DP, Tsitsiklis JN. Parallel and distributed computation: Numerical methods. Engl. Cliffs, N. J: Prentice-Hall; 1989, (can be downloaded from the author's website).
[67] Bertsekas DP. The auction algorithm for assignment and other network flow problems: A tutorial. Interfaces 1990;20:133–49.

[68] Bertsekas DP. Auction algorithms for network flow problems: A tutorial introduction. Comput Optim Appl 1992;1:7–66.
[69] Bertsekas DP. The auction algorithm: A distributed relaxation method for the assignment problem. Ann Oper Res 1988;14:105–23.
[70] Karzanov AV. Determining the maximal flow in a network with the method of preflows. Sov Math Dokl 1974;15:1277–80.
[71] Shiloach Y, Vishkin U. An $O(n^2 \log n)$ parallel max-flow algorithm. J Algorithms 1982;3:128–46.
[72] Goldberg AV, Tarjan RE. A new approach to the maximum flow problem. In: Proc. 18th ACM STOC. 1986, p. 136–46.
[73] Goldberg AV, Tarjan RE. Solving minimum cost flow problems by successive approximation. Math Oper Res 1990;15:430–66.
[74] Bertsekas DP, Eckstein J. Distributed asynchronous relaxation methods for linear network flow problems. IFAC Proc 1987;20:103–14.
[75] Ahuja RK, Orlin JB. A fast and simple algorithm for the maximum flow problem. Oper Res 1989;37:748–59.
[76] Cheriyan J, Maheshwari SN. Analysis of preflow push algorithms for maximum network flow. SIAM J Comput 1989;18:1057–86.
[77] Orlin JB, Ahuja RK. New scaling algorithms for the assignment and minimum mean cycle problems. Math Program 1992;54:41–56.
[78] Khosla M, Anand A. Revisiting the auction algorithm for weighted bipartite perfect matchings. 2021, arXiv preprint arXiv:2101.07155.
[79] Bernard F. Fast linear assignment problem using auction algorithm, MATLAB central file exchange. 2023, https://www.mathworks.com/matlabcentral/fileexchange/48448-fast-linear-assignment-problem-using-auction-algorithm-mex.
[80] Bertsekas DP. Rollout, policy iteration, and distributed reinforcement learning. Belmont, MA: Athena Scientific; 2020.
[81] Bertsekas DP. Constrained multiagent rollout and multidimensional assignment with the auction algorithm. 2020, arXiv preprint, arXiv:2002.07407.
[82] Blackman SS. Multi-target tracking with radar applications. Dehdam, MA: Artech House; 1986.
[83] Bar-Shalom Y, Fortman TE. Tracking and data association. N. Y: Academic Press; 1988.
[84] Bar-Shalom Y. Multitarget-multisensor tracking: Advanced applications. Norwood, MA: Artech House; 1990.
[85] Castañon DA. New assignment algorithms for data association. In: Signal and data processing of small targets, vol. 1698. 1992, p. 313–23.
[86] Pattipati KR, Deb S, Bar-Shalom Y, Washburn RB. A new relaxation algorithm and passive sensor data association. IEEE Trans Automat Control 1992;37:198–213.
[87] Poore AB. Multidimensional assignment formulation of data association problems arising from multitarget tracking and multisensor data fusion. Comput Optim Appl 1994;3:27–57.
[88] Poore AB, Robertson AJA. New Lagrangian relaxation based algorithm for a class of multidimensional assignment problems. Comput Optim Appl 1997;8:129–50.
[89] Popp RL, Pattipati KR, Bar-Shalom Y. m-Best SD assignment algorithm with application to multitarget tracking. IEEE Trans Aerosp Electron Syst 2001;37:22–39.
[90] Emami P, Pardalos PM, Elefteriadou L, Ranka S. Machine learning methods for data association in multi-object tracking. ACM Comput Surv 2020;53:1–34.
[91] Bertsekas DP, Castañon DA. Parallel synchronous and asynchronous implementations of the auction algorithm. Parallel Comput 1991;17:707–32.
[92] Wein J, Zenios SA. On the massively parallel solution of the assignment problem. J Parallel Distrib Comput 1991;13:228–36.
[93] Amini MM. Vectorization of an auction algorithm for linear cost assignment problem. Comput Ind Eng 1994;26:141–9.
[94] Bertsekas DP, Castañon DA, Eckstein J, Zenios S. Parallel computing in network optimization. In: Ball MO, Magnanti TL, Monma CL, Nemhauser GL, editors. In: Handbooks in OR and MS, vol. 7, Amsterdam: North-Holland; 1995, p. 331–99.
[95] Beraldi P, Guerriero F, Musmanno R. Efficient parallel algorithms for the minimum cost flow problem. J Optim Theory Appl 1997;95:501–30.
[96] Beraldi P, Guerriero F, Musmanno R. Parallel algorithms for solving the convex minimum cost flow problem. Comput Optim Appl 2001;18:175–90.
[97] Beraldi P, Guerriero F. A parallel asynchronous implementation of the ϵ-relaxation method for the linear minimum cost flow problem. Parallel Comput 1997;23:1021–44.
[98] Zavlanos MM, Spesivtsev L, Pappas GJ. Distributed auction algorithm for the assignment problem. In: Proc. 47th IEEE conference on decision and control. 2008. A, p. 1212–7.
[99] Bus L, Tvrdk P. Towards auction algorithms for large dense assignment problems. Comput Optim Appl 2009;43:411–36.
[100] Sathe M, Schenk O, Burkhart H. An auction-based weighted matching implementation on massively parallel architectures. Parallel Comput 2012;38:595–614.
[101] Nascimento CN, Jamel FS, Sena AC. A hybrid parallel algorithm for the auction algorithm in multicore systems. In: 2016 international symposium on computer architecture and high performance computing workshops. 2016, p. 73–8.
[102] Sena AC, Silva MN, Nascimento AP. An efficient vectorized auction algorithm for many-core and multicore architectures. In: Latin American high performance computing conference. Springer; 2021, p. 76–90.
[103] Bertsekas DP. An auction/sequential shortest path algorithm for the minimum cost network flow problem. Report LIDS-P-2146, MIT; 1995.
[104] Bertsekas DP. Auction algorithms for path planning, network transport, and reinforcement learning. Arizona State University/SCAI report, 2022, arXiv:22207.09588.
[105] Agrawal G, Bertsekas D, Liu H. Auction-based learning for question answering over knowledge graphs. Information 2023;14(336). http://dx.doi.org/10.3390/info14060336.
[106] Bertsekas DP. An auction algorithm for the max-flow problem. J Optim Theory Appl 1995;87:69–101.
[107] Bertsekas DP, Castañon DA, Tsaknakis H. Reverse auction and the solution of inequality constrained assignment problems. SIAM J Optim 1993;3:268–99.
[108] Bertsekas DP, Castañon DA. The auction algorithm for the transportation problem. Ann Oper Res 1989;20:67–96.
[109] Bertsekas DP. An auction algorithm for shortest paths. SIAM J Optim 1991;1:425–47.
[110] Bertsekas DP. Distributed asynchronous relaxation methods for linear network flow problems. In: 1986 25th IEEE conference on decision and control, vol. 210. 1986, p. 1–2106.
[111] Bertsekas DP, Castañon DA. A generic auction algorithm for the minimum cost network flow problem. Comput Optim Appl 1993;2:229–60.
[112] Bertsekas DP, Castañon DA. Parallel asynchronous hungarian methods for the assignment problem. ORSA J Comput 1993;5:261–74.
[113] Bertsekas DP, Castañon DA. Parallel primal-dual methods for the minimum cost flow problem. Comput Optim Appl 1993;2:317–36.
[114] Bertsekas DP. A course in reinforcement learning. Belmont, MA: Athena Scientific; 2023.
[115] Lee M, Xiong Y, Yu G, Li GY. Deep neural networks for linear sum assignment problems. IEEE Wirel Commun Lett 2018;7:962–5.