Proceedings of the
47th IEEE Conference on Decision and Control
Cancun, Mexico, Dec. 9-11, 2008

A Distributed Auction Algorithm for the Assignment Problem

Michael M. Zavlanos, Leonid Spesivtsev and George J. Pappas

Abstract— **The assignment problem constitutes one of the fundamental problems in the context of linear programming. Besides its theoretical significance, its frequent appearance in the areas of distributed control and facility allocation, where the problems' size and the cost for global computation and information can be highly prohibitive, gives rise to the need for local solutions that dynamically assign distinct agents to distinct tasks, while maximizing the total assignment benefit. In this paper, we consider the linear assignment problem in the context of networked systems, where the main challenge is dealing with the lack of global information due to the limited communication capabilities of the agents. We address this challenge by means of a distributed auction algorithm, where the agents are able to bid for the task to which they wish to be assigned. The desired assignment relies on an appropriate selection of bids that determine the prices of the tasks and render them more or less *attractive* for the agents to bid for. Up to date pricing information, necessary for accurate bidding, can be obtained in a multi-hop fashion by means of local communication between adjacent agents. Our algorithm is an extension to the parallel auction algorithm proposed by Bertsekas *et al* to the case where only local information is available and it is shown to always converge to an assignment that maximizes the total assignment benefit within a linear approximation of the optimal one.**

I. INTRODUCTION

Given two sets consisting of agents and tasks, respectively, the linear assignment problem searches for a one-to-one matching between the agents and the tasks so that the total assignment benefit is maximized. In this paper, we investigate the linear assignment problem in the context of networked systems, where computation and memory resources are distributed among a set of agents with limited communication capabilities seeking an assignment with a desired set of tasks.

The linear assignment problem is fundamental in combinatorial optimization due to its underlying linear programming structure. It can be shown that the well known class of linear network flow problems can be reduced to the linear assignment problem by an appropriate transformation [1]. Moreover, the linear assignment problem often appears as a subproblem in more complex problems such as the traveling salesman problem [2]. Besides, however, its theoretical significance, equally important is its wide applicability in research areas ranging from facility allocation to distributed robotics. In distributed robotics, in particular, the linear assignment problem has recently received considerable attention due to applications involving task or target allocation [3]–[6] and formation stabilization [7]–[16].

Michael M. Zavlanos, Leonid Spesivtsev and George J. Pappas are with GRASP Laboratory, School of Engineering and Applied Science, University of Pennsylvania, Philadelphia, PA 19104, USA {zavlanos,spesiv,pappasg}@grasp.upenn.edu

Due to its theoretical and practical significance, various techniques have been proposed for obtaining an optimal solution. Most rely on iterative improvement of some cost function [17]–[20], while Kuhn's Hungarian algorithm [21] was the first method specifically designed for that problem. All above techniques require a single central processor that handles all computation and information in the system. The need, however, for more efficient and reliable algorithms, has recently lead to decentralized approaches, where computation and ideally also information is distributed among multiple parallel processing units. One such approach is the *auction algorithm* proposed in [22], [23] that based on an appropriate choice of bids, determines the prices of the tasks and renders them more or less attractive for the agents to bid for. Although, the assignment benefit does not necessarily increase monotonically, it can be shown that eventually an optimal assignment is discovered.

Despite the fact that computation in the auction algorithm [23] is distributed over multiple agents, a shared memory repository containing the task prices, where all agents have access, is required. This practically leads to a complete communication topology. In the case of networked systems, however, consisting of possibly mobile agents with power constraints and limited communication capabilities, the underlying network topologies are in principle *dynamic* and not complete. Hence, we need to extend the parallel auction algorithm proposed in [23], to the case where only local communication between adjacent agents is available. As in [23], the assignment process relies on an appropriate selection of bids that determine the task prices, however, in the current framework every agent locally stores, possibly outdated, estimates of the task prices, which it updates using nearest neighbor agreement protocols. We show that the network topology may affect the algorithm performance which, however, is guaranteed to converge to an assignment with total assignment benefit within a linear approximation of the optimal one. We also argue that in mobile robotics applications, where neither shared memory nor central computation is available, our approach is more natural.

The rest of this paper is organized as follows. In Section II we define the linear assignment problem in the context of networked systems, while in Section III we propose a distributed auction algorithm for the assignment problem and discuss its convergence and complexity properties. Finally, in Section IV, we illustrate our approach for different network topologies and discuss the resulting performance.

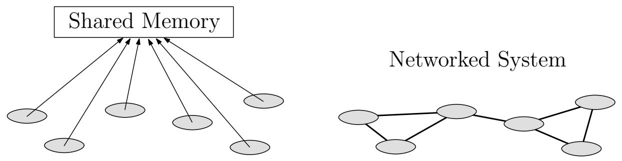


Fig. 1. *Shared memory* (left) versus *networked* (right) systems. Both systems consist of parallel independent processing units (agents) resulting in asynchronous decentralized computation. Information, however, in shared memory systems is global due to a central memory repository containing the system's global variables where all agents have read and write access, whereas in networked systems is local due to the agents' local memory (where local variables are stored) and their limited communication capabilities that allow information exchange with nearest neighbors only. Furthermore, shared memory systems are typically associated with static communication topologies, while communication topologies in the case of networked systems can, in general, be dynamic.

II. Problem Formulation

Consider a network of n agents with integrated wireless communication capabilities and denote by (i, j) a communication link between agents i and j. We assume that communication links between the agents can be enabled and disabled in time, due to either agent mobility, or power constraints. Such networks give rise to the notion of a *dynamic* graph $\mathcal{G}(t) = (\mathcal{V}, \mathcal{E}(t))$, where $\mathcal{V} = \{1, \dots, n\}$ consists the set of vertices indexed by the set of agents and $\mathcal{E}(t) = \{(i, j) \mid i, j \in \mathcal{V}\}$ denotes a time varying set of links. We assume bidirectional communication links among the agents and so $(i, j) \in \mathcal{E}(t)$ if and only if $(j, i) \in \mathcal{E}(t)$. Such graphs are called *undirected* and consist the main focus of this paper. Any vertices i and j of an undirected graph $\mathcal{G}(t)$ that are joined by a link $(i, j) \in \mathcal{E}(t)$, are called *adjacent* or *neighbors* at time t. Hence, we can define the set of neighbors of agent i at time t, by $\mathcal{N}_i(t) = \{j \in \mathcal{V} \mid (i, j) \in \mathcal{E}(t)\}$. A topological invariant of graphs that is of great importance for the purposes of this work is graph *connectivity*.

Definition 2.1 (Graph Connectivity): We say that a dynamic graph $\mathcal{G}(t)$ is connected at time t if there exists a path, i.e., a sequence of distinct vertices such that consecutive vertices are adjacent, between any two vertices in $\mathcal{G}(t)$.

Given the dynamic network $\mathcal{G}(t)$ consisting of n agents described above, let $m \geq n$ denote a number of tasks that need to be accomplished by the agents and define the injective map $\alpha : \{1, \dots, n\} \to \{1, \dots, m\}$ such that $\alpha(i) = j$ if and only if task j has been assigned to agent i. Denote, further, by $\beta_{ij} \in \mathbb{R}$ the benefit of assigning task j to agent i. This can be a function of the distance that agent i needs to travel in order to acquire task j, the power that is required to fulfill task j, or even the time required to fulfill that task. Then, the objective investigated in this paper can be stated as follows.

Problem 1 (Distributed Assignment): Given a connected in time dynamic network $\mathcal{G}(t)$ consisting of n agents, $m \geq n$ tasks and a set of nm benefits β_{ij} associated with assigning agent i to task j, determine distributed control laws that assign distinct agents to distinct tasks, such that the total assignment benefit $\sum_{i=1}^{n} \beta_{i\alpha(i)}$ is maximized.

Since, the assignment of the agents to tasks is not provided a priori, it needs to be determined *dynamically*. We achieve this goal by defining a distributed market where the agents are able to bid for the task to which they wish to be assigned. Unlike prior work that assumes either a single central auctioneer [22] or multiple parallel processing units (corresponding to the agents) composed in an asynchronous shared memory system with global information [23], we propose a distributed control framework, where every agent independently and using only *local* information, is able to determine a task to be assigned to (Fig. 1). The desired assignment relies on an appropriate selection of bids that determine the prices of the tasks and render them more or less *attractive* for the agents to bid for. Accurate pricing information, necessary for correct bidding, can be obtained in a multi-hop fashion by means of local communication between adjacent agents. Our approach is described in Section III and consists an extension to the parallel auction algorithm [23], proposed by Bertsekas *et al*, to the case where only local information is available.

III. Distributed Auction Algorithm

As in [22], let every task j have a price $p_j(t) \geq 0$ at time t, which every agent i that gets assigned to it has to pay. Then, the net value of task j for agent i is $\beta_{ij} - p_j(t)$ and every agent i would like to be assigned to a task j that provides it with a maximum net value

$$\beta_{ij} - p_j(t) = \max_{1 \leq k \leq m} \{\beta_{ik} - p_k(t)\}. \tag{1}$$

If (1) is satisfied for all agents i we say that the assignment and the set of prices are at *equilibrium*. Equilibrium assignments are fundamental in the study of the assignment problem since, they correspond to maximum total benefit, while the corresponding sets of prices solve the associated dual optimization problem [24].

As shown in [22], designing a market that guarantees an equilibrium assignment is not straightforward, due to the possibility of cycles in the algorithm, resulting from several agents requesting to be assigned to a smaller number of equally desirable tasks without, however, raising their prices. For this, we employ the notion of an *almost equilibrium* assignment and set of prices, introduced in [22]. This notion of an equilibrium is motivated by real auctions, where every bid for a task must raise its price by a minimum positive increment and the agents must take risks to win their preferred tasks. In particular, we define an *almost equilibrium* assignment and set of prices at time t when the net value for every agent i assigned to task j is within a constant $\epsilon > 0$ of being maximal, i.e., when

$$\beta_{ij} - p_j(t) \geq \max_{1 \leq k \leq m} \{\beta_{ik} - p_k(t)\} - \epsilon \tag{2}$$

for all agents i. Condition (2) in the context of the auction algorithm [22] is known as *ϵ-complementary slackness* and for $\epsilon = 0$ reduces to the ordinary complementary slackness condition (1).

The rest of this section is devoted in describing a distributed auction algorithm over a dynamic network of agents

Algorithm 1 Auction Iteration for Agent i

```
Require: An assignment $\alpha_i(t) \in \{1,\dots,m\}$ and a set of prices $p_{ij}(t) \geq 0$ and highest bidders $b_{ij}(t) \in \mathbb{N}$ for all tasks $1 \leq j \leq m$;
1: Update the prices of all tasks and the corresponding highest bidders by
   $p_{ij}(t+1) := \max_{k\in\mathcal{N}_i(t)}\{p_{ij}(t), p_{kj}(t)\}$ and
   $b_{ij}(t+1) := \max_{k\in\text{argmax}_{z\in\mathcal{N}_i(t)}\{p_{ij}(t),p_{zj}(t)\}}\{b_{kj}(t)\}$;
2: if $p_{i\alpha_i(t)}(t) \leq p_{i\alpha_i(t)}(t+1)$ and $b_{i\alpha_i(t)}(t+1) \neq i$ then
3:    Update the assignment by
      $\alpha_i(t+1) \in \text{argmax}_{1\leq k\leq m}\{\beta_{ik} - p_{ik}(t+1)\}$;
4:    Set $b_{i\alpha_i(t+1)}(t+1) := i$ and increase the price for task $\alpha_i(t+1)$ by $p_{i\alpha_i(t+1)}(t+1) := p_{i\alpha_i(t+1)}(t) + \gamma_i$, where $\gamma_i \geq \epsilon$ is according to equations (3)-(5);
5: else
6:    Remain assigned to task $\alpha_i(t)$, i.e., $\alpha_i(t+1) := \alpha_i(t)$;
7: end if
```

and showing that it always converges to an almost equilibrium assignment. In particular, let $\mathcal{G}(t)$ denote a dynamic network consisting of n agents and let $\alpha_i(t) \in \{1,\dots,m\}$ denote the assignment status of agent i at time t, such that $\alpha_i(t) = j$ if agent i is assigned to task j. Let further $p_{ij}(t) \geq 0$ denote the price that agent i needs to pay in order to be assigned to task j at time t and $b_{ij}(t) \in \mathbb{N}$ denote the *largest-index bidder* among the possibly multiple (due to ties) highest bidders for task j at time t.[1]

Given the above notation, a single iteration of the distributed auction algorithm for agent i is described in Algorithm 1. In particular, given a set of prices $\{p_{ij}(t)\}_{j=1}^m$ at time t, an assignment $\alpha_i(t)$ that currently provides agent i with the best net value, i.e., $\alpha_i(t) \in \text{argmax}_{1\leq k\leq m}\{\beta_{ik} - p_{ik}(t)\}$, and a set of highest bidders with the largest index $\{b_{ij}(t)\}_{j=1}^m$, such that $b_{i\alpha_i(t)}(t) = i$, agent i updates the prices and highest bidders for all tasks j using local *maximum-price* and *maximum-index* update protocols, respectively (line 1, Alg. 1). Such updates guarantee that, for any sequence $\mathcal{G}(t)$ of connected networks, every agent will eventually receive the up-to-date maximum price

$$p_j(t) \triangleq \max_{1\leq k\leq n}\{p_{kj}(t)\}$$

of all tasks j as well as the corresponding highest bidders with the largest index. In other words, although time delays in the network due to multi-hop information propagation may result in non-adjacent agents using *outdated* task prices and bidding lower for expensive tasks, eventually up-to-date information is received resulting in accurate bidding (Fig. 2).

The second and final step of an iteration of Algorithm 1 consists of checking whether the price $p_{i\alpha_i(t)}(t)$ of the current assignment $\alpha_i(t)$ of agent i has been increased by

[1]Unlike shared memory systems where the global task prices $p_j(t)$ are stored in a common memory repository to which all agents have access [23], in the proposed networked system every agent i has local copies of the task prices $p_{ij}(t)$ which it updates using information from its neighbors (Fig. 1). This lack of global information, may result in *ties* in the bids, for which a *tie breaking* mechanism, captured by $b_{ij}(t)$, needs to be introduced.

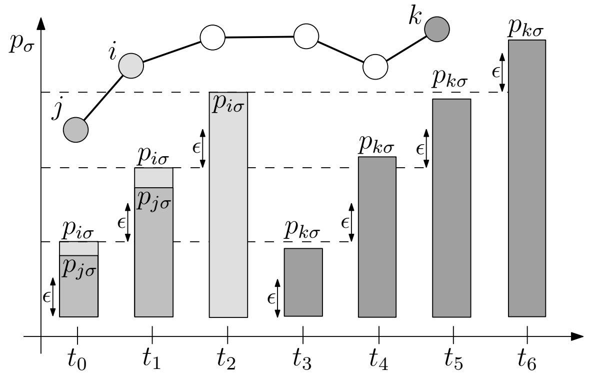


Fig. 2. Propagation of pricing information in a network of six agents. Agents i, j and k desire to be assigned to task σ and bid for this task at time instants t_0 through t_6 corresponding to consecutive communication cycles. Assume that task σ is initially *attractive* for agents i and j and it remains attractive for both at time t_1 after they have exchanged prices $p_{i\sigma}(t_0)$ and $p_{j\sigma}(t_0)$, respectively. During the next communication cycle (time t_2), agent i receives $p_{j\sigma}(t_1)$ from agent j and bids higher for task σ, which is no longer attractive for agent j. At time t_3 agent k is not yet aware of the pricing sequence $p_{i\sigma}(t_0)$, $p_{i\sigma}(t_1)$ and $p_{i\sigma}(t_2)$ and initializes a low bid for task σ. However, during subsequent time instants t_4, t_5 and t_6, agent k receives prices $p_{i\sigma}(t_0)$, $p_{i\sigma}(t_1)$ or $p_{i\sigma}(t_2)$, respectively, and increases its bid for task σ. Note that lack of global pricing information results in agent k initially placing unreasonably low bids. However, eventually a correct bid will be placed due to the connected network structure.

other agents in the network or whether a larger-indexed agent has placed an equal bid, in which case there is a tie for task $\alpha_i(t)$ (line 2, Alg. 1).[2] If any of the previous statements is true, assignment $\alpha_i(t)$ may no longer be at equilibrium and agent i needs to select a new assignment $\alpha_i(t+1)$ using the updated prices (line 3, Alg. 1) and increase the price $p_{i\alpha_i(t+1)}$ of this task by

$$\gamma_i \triangleq v_i - w_i + \epsilon, \tag{3}$$

where

$$v_i \triangleq \max_{1\leq j\leq m}\{\beta_{ij} - p_{ij}(t)\} \tag{4}$$

corresponds to the best net value available to agent i,

$$w_i \triangleq \max_{j\neq\alpha_i(t+1)}\{\beta_{ij} - p_{ij}(t)\} \tag{5}$$

corresponds to the second best net value available to agent i and $\epsilon > 0$ indicates a minimum bid increment (line 4, Alg. 1). Clearly,

$$\max_{1\leq j\leq m}\{\beta_{ij} - p_{ij}(t+1)\} - \epsilon \leq \beta_{i\alpha_i(t+1)} - p_{i\alpha_i(t+1)}(t+1),$$

which implies that the new assignment and set of prices are almost at equilibrium, with respect to the *partial* knowledge of prices that agent i has. Note that, with respect to this partial knowledge of prices, $p_{i\alpha_i(t+1)}(t+1)$ is now the highest price for task $\alpha_i(t+1)$ and so agent i can also update $b_{i\alpha_i(t+1)}(t+1)$, accordingly (line 4, Alg. 1). Note, finally, that Algorithm 1 does not require any particular initialization of the prices $p_{ij}(0)$. In other words, the necessary almost at

[2]Note that if $p_{i\alpha_i(t)}(t) < p_{i\alpha_i(t)}(t+1)$ then clearly $b_{i\alpha_i(t)}(t+1) \neq i$. Hence, the condition $b_{i\alpha_i(t)}(t+1) \neq i$ only affects ties on the prices, where it acts as a tie breaker due to the *maximum-index* update rule (line 1, Alg. 1). Clearly, the largest-index agent wins all ties.

equilibrium initial assignment $\alpha_i(0)$ and corresponding set of highest bidders $b_{ij}(0)$, with $b_{i\alpha_i(0)}(0) = i$, can be generated using any set of initial prices.

Hence, we can define the *distributed auction algorithm* for the assignment problem by the integration of n copies of Algorithm 1 run by each one of the agents in the network. The following result shows that the proposed algorithm always terminates with a correct assignment.

Proposition 3.1: Given a connected in time dynamic network $\mathcal{G}(t)$ consisting of n agents and a set of $m \geq n$ tasks, the distributed auction algorithm terminates in a finite number of iterations with an assignment and a set of prices that are almost at equilibrium.

Proof: Note first that, due to the networked structure of the system which imposes multi-hop communication patterns among the agents, for every task and every time instant there exist agents that are not yet informed of its actual price and others that are (Fig. 2). We call these agents *uninformed* and *informed*, respectively, and note that due to the maximum-price update rule and the connected structure of the network, every uninformed agent becomes informed in a *finite* number of communication cycles that depends on its distance (in number of links) to the closest informed agent.[3] This implies that uninformed agents may be placing low bids for expensive tasks, however, eventually they will become informed and bid correctly for an *attractive* task. With this observation, we can disregard all bids made by uninformed agents (which are finite due to the finite number of communication cycles until an agent becomes informed) and only consider bids by informed agents that increase the actual price of every task. Hence, we only need to show that every task can only receive a finite number of such bids.

Our argument is along the lines of [22]: Observe that every informed agent that is assigned to a task that has already received a bid, attains the maximum possible net value from this assignment. The reason for this is that the maximum net value is attained by any informed agent just after acquiring the task and remains maximum for as long as the agent remains informed and holds the task (since the other task prices can not decrease in the course of the algorithm). Note also that whenever m bids are placed for a task by any number of informed agents, its price must increase by at least $m\epsilon$. Thus, for sufficiently large m, the task will become *expensive* enough to be considered as less *attractive* compared to other tasks that have not yet received any bids. It follows that there is a limited number of bids that any task can receive by informed agents, while there still exist tasks that have not yet received any bids. Therefore, the auction will continue until all tasks have received at least one bid by an informed agent (at which point all agents are informed) and will terminate with all agents obtaining their maximum possible net value. ■

Observe that Theorem 3.3 also provides a *termination condition* for the distributed auction algorithm, namely that every task should have received at least one bid. This condition clearly holds when the task prices stop changing and in order to account for the multi-hop information propagation in the network, we may define termination of the algorithm as the time instant when the task prices for all agents remain unchanged for at least Δ communication rounds, where $\Delta \leq n-1$ indicates the maximum *network diameter*, i.e., the maximum length path between any two agents in the network. The following result provides an upper bound on the number of iterations of the distributed auction algorithm.

Proposition 3.2: The distributed auction algorithm terminates in $O\big(\Delta n^2 \lceil \frac{\max_{i,j}\{\beta_{ij}\} - \min_{i,j}\{\beta_{ij}\}}{\epsilon} \rceil\big)$ iterations.

Proof: In order to maximize the total number of iterations of the algorithm, we construct a worst case scenario where all agents persistently place minimum bid increments of size $\epsilon > 0$ on every single task (until it is no longer attractive), delaying thus the assignment process. To achieve this goal, let $\delta > 0$ and $M_j > 0$, $j = 1, \ldots, m$, such that the benefits β_{ij} are distributed according to

$$\begin{cases} \max_{1\leq i\leq n}\{\beta_{ij}\} - \min_{1\leq i\leq n}\{\beta_{ij}\} < \delta \\ \min_{1\leq i\leq n}\{\beta_{ij}\} - \max_{1\leq i\leq n}\{\beta_{i(j-1)}\} = M_j \end{cases},$$

for all $j = 1\ldots, m$. Then, task n is initially the most attractive for all agents and for sufficiently small $\delta > 0$ it will remain so until its price is increased by at least M_n. This requires at least $\lceil \frac{M_n}{\epsilon} \rceil$ bids on that task by every agent and results in a total of $n\lceil \frac{M_n}{\epsilon} \rceil$ iterations of the algorithm. Once the price of task n has been increased by at least M_n, task $n-1$ becomes also attractive and starts receiving bids by the agents. Clearly, tasks n and $n-1$ remain the only attractive tasks for at least $2\lceil \frac{M_{n-1}}{\epsilon} \rceil$ more iterations of the algorithm by every one of the n agents, resulting in an increase in their price by at least M_{n-1} and a total number of $2n\lceil \frac{M_{n-1}}{\epsilon} \rceil$ iterations. Similarly, the next task to become attractive and start receiving bids is task $n-2$ and as before, tasks n, $n-1$ and $n-2$ remain the only attractive tasks for at least $3n\lceil \frac{M_{n-2}}{\epsilon} \rceil$ more iterations of the algorithm. Proceeding in the same fashion and summing up the total numbers of iterations for each stage of the assignment process, we get

$$n \sum_{j=1}^{n} j \left\lceil \frac{M_{n-j+1}}{\epsilon} \right\rceil \leq n^2 \left\lceil \frac{\max_{i,j}\{\beta_{ij}\} - \min_{i,j}\{\beta_{ij}\}}{\epsilon} \right\rceil.$$

Observe that this number of iterations needs to be augmented by the number of communication rounds required to propagate each bid in the network. In the worst case, this requires Δ communication rounds for every iteration of the algorithm, which completes the proof. Note that, the proposed scenario forces the agents to keep bidding on specific tasks and hence, delays the assignment process and maximizes the total number of iterations, as desired. Note also, that for a complete network topology, $\Delta = 1$ and the upper bound on the number of iterations of the distributed auction algorithm reduces to the one obtained in [22]. ■

The following result characterizes the total benefit achieved by the distributed auction algorithm in terms of

[3]Note that the number of communication cycles is finite since, as it will be shown next, there can not exist informed agents that continuously bid for the same task.

the optimal one. In particular, since the networked structure of the system only affects the underlying information pattern and not the nature of the assignment procedure, any results on optimality of solutions obtained by the auction algorithm [22] apply in the case of the distributed auction algorithm as well. Hence, we have the following theorem.

Theorem 3.3 (Adopted from [22]): The final assignment $\alpha : \{1,\dots,n\} \to \{1,\dots,m\}$ with $\alpha(i) \triangleq \alpha_i$ that is obtained by the distributed auction algorithm is within $n\epsilon$ of maximizing the the total assignment benefit.

Proof: Let $\epsilon > 0$ and note that the total benefit of *any* assignment $\alpha \in \{1,\dots,m\}$ satisfies

$$\sum_{i=1}^{n} \beta_{i\alpha(i)} \leq \sum_{j=1}^{m} p_j + \sum_{i=1}^{n} \max_{1\leq j\leq m} \{\beta_{ij} - p_j\},$$

for any set of prices $\{p_j\}_{j=1}^m$, since the second term on the right-hand-side is no less than $\sum_{i=1}^n(\beta_{i\alpha(i)} - p_{\alpha(i)})$, while the first term is equal to $\sum_{i=1}^n p_{\alpha(i)}$. Hence, $A^\star \leq D^\star$, where

$$A^\star \triangleq \max_{\alpha(i),\, i=1,\dots,n} \sum_{i=1}^{n} \beta_{i\alpha(i)}$$

is the optimal total assignment benefit and

$$D^\star \triangleq \min_{p_j,\, j=1,\dots,m} \sum_{j=1}^{m} p_j + \sum_{i=1}^{n} \max_{1\leq j\leq m} \{\beta_{ij} - p_j\}.$$

Since the final assignment and set of prices obtained by the distributed auction algorithm are almost at equilibrium, we also have that $\beta_{i\alpha(i)} - p_{\alpha(i)} \geq \max_{1\leq j\leq m}\{\beta_{ij} - p_j\} - \epsilon$, which implies that

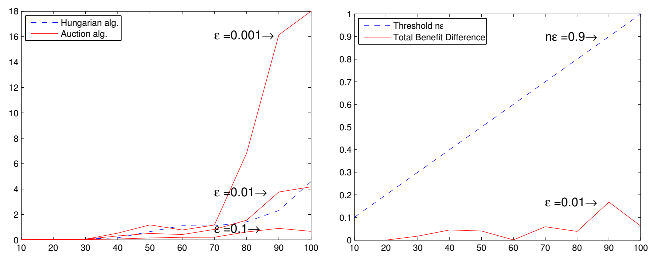
$$\begin{aligned} D^\star &\leq \sum_{i=1}^{n} \Big(p_{\alpha(i)} + \max_{1\leq j\leq m} \{\beta_{ij} - p_j\}\Big) \\ &\leq \sum_{i=1}^{n} \beta_{i\alpha(i)} + n\epsilon \leq A^\star + n\epsilon. \end{aligned}$$

Since $A^\star \leq D^\star$, it follows that the total assignment benefit $\sum_{i=1}^n \beta_{i\alpha(i)}$ is within $n\epsilon$ of the optimal value $A^\star$. ■

Theorem 3.3 shows that for sufficiently small ϵ, the final assignment is *almost optimal*. In particular, if all benefits β_{ij} are integers, the total benefit $\sum_{i=1}^n \beta_{i\alpha(i)}$ for any assignment α is also integer. Hence, if $n\epsilon < 1$, an assignment that is within $n\epsilon$ of being optimal, must be optimal. We conclude that if $\epsilon < 1/n$ and all benefits are integers, the final assignment obtained by the distributed auction algorithm is optimal. This is a straightforward extension of a similar observation made in [22].

IV. Algorithm Performance & the Effect of the Network Topology

In this section we illustrate the proposed distributed auction algorithm for different values of the parameter $\epsilon > 0$ and different, but fixed, network topologies $\mathcal{G}(t) = \mathcal{G}$. In particular, for different problem sizes n and same number of tasks ($m = n$), we randomly generated benefits β_{ij} from a uniform distribution on the unit interval and compared the running time and final assignment benefit $\sum_{i=1}^n \beta_{i\alpha(i)}$ obtained by the distributed auction algorithm with the optimal one $\sum_{i=1}^n \beta_{i\alpha^\star(i)}$ obtained by the Hungarian algorithm [21]. The communication topologies considered were the minimally connected *line topology*, where every agent is connected to at most two other agents (Fig. 2), the *complete topology*, where every agent is connected to all other agents in the network, and a *random topology*, where every agent is connected to 50% of the other agents so that the resulting network is connected. All algorithms were implemented in MATLAB and run on an Intel Core 2 Duo 2.40 GHz processor with 4 GB RAM.



(a) Time (sec) vs. problem size n for different values of the parameter $\epsilon = .1, .01, .001$.

(b) Difference in final assignment benefit (Hungarian minus Auction) vs. problem size n for parameter $\epsilon = .01$.

Fig. 3. Comparison of the performance of the distributed auction algorithm and the Hungarian algorithm for different problem sizes n and a complete network topology. Observe that small values of ϵ result in slow convergence of the auction algorithm, while for large ϵ the auction algorithm becomes faster that the Hungarian. Note also that the final assignment benefit obtained by the auction algorithm is within $n\epsilon$ of the optimal one (Theorem 3.3).

Fig. 3 compares the performance of the distributed auction algorithm with the Hungarian algorithm for different problem sizes n, parameters ϵ and the complete network topology. Observe that the larger ϵ is, the faster the distributed auction algorithm becomes (Fig. 3(a)). In particular, for sufficiently large values of ϵ, the auction algorithm becomes significantly faster than the Hungarian algorithm. This is a reasonable observation given the upper bound on the running time of the algorithm obtained in Proposition 3.2. Although large values of ϵ result in faster convergence, they may also result in worse performance with respect to the final assignment benefit $\sum_{i=1}^n \beta_{i\alpha(i)}$. Theorem 3.3, however, guarantees that this final assignment benefit is always within $n\epsilon$ of the optimal one. This is illustrated in Fig. 3(b), where the

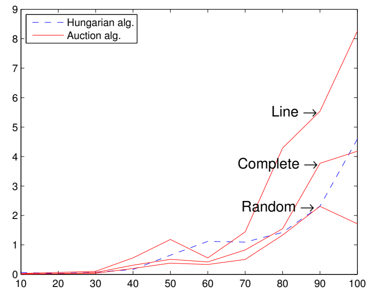
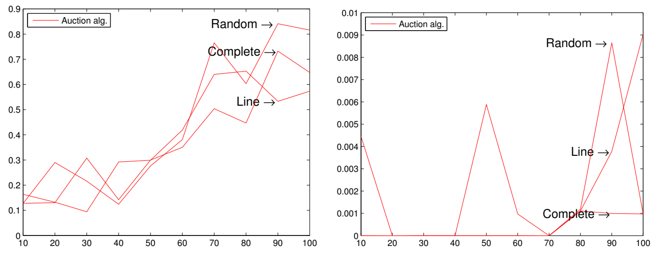


Fig. 4. Time (sec) vs. problem size n for different network topologies and $\epsilon = .01$. Note the slow convergence of the line topology due to the corresponding maximum network diameter $n-1$ that dominates complexity.



(a) Difference in final assignment benefit (Hungarian minus Auction) vs. problem size n for parameter $\epsilon = .1$.

(b) Difference in final assignment benefit (Hungarian minus Auction) vs. problem size n for parameter $\epsilon = .001$.

Fig. 5. Comparison of the performance of the distributed auction algorithm for different network topologies and parameters ϵ. Observe that the final assignment depends on the topology of the network, however, for sufficiently small ϵ, the auction algorithm returns an almost optimal solution (Theorem 3.3) and this effect disappears.

difference in the final assignment benefit $\sum_{i=1}^n \beta_{i\alpha^\star(i)} - \sum_{i=1}^n \beta_{i\alpha(i)}$ obtained by the Hungarian algorithm and the auction algorithm, respectively, is plotted as a function of the problem size n for $\epsilon = .01$.

The effect of the network topology on the performance of the auction algorithm is considered in Figs. 4 and 5. In particular, Fig. 4 shows how the network topology affects the running time of the auction algorithm. Observe that sparse topologies (line topology) are the slowest ones to converge, as expected by complexity bounds obtained by Proposition 3.2. On the other hand, the final assignments obtained for the same problem on different communication topologies are not necessarily the same (Fig. 5). This is a reasonable observation given the fact that the network topology affects the order in which the agents bid for attractive tasks and for large bid increments, this order may render tasks unattractive (Fig. 5(a)). Note, however, that the smaller ϵ becomes, the smaller the bid increments are and the less the network topology affects the assignment (Fig. 5(b)).

V. Conclusions

In this paper, we considered the linear assignment problem in the context of networked systems, where the main challenge was dealing with the lack of global information due to the limited communication capabilities of the agents. We addressed this challenge by means of a distributed auction algorithm, where every agent was able to bid for any desired task. The final assignment was obtained through an appropriate choice of bids that determined the prices of the tasks and rendered them more or less attractive for the agents to bid for. Our algorithm consisted an extension to the parallel auction algorithm proposed by Bertsekas *et al*, to the case where no shared memory is available and the agents are required to store locally all pricing information and update it in a multi-hop fashion using nearest neighbor agreement protocols. We showed that our algorithm converges to an assignment that maximizes the total assignment benefit within a linear approximation of the optimal one and discussed the effect of the underlying network topology on its performance.

References

[1] C. H. Papadimitriou and K. Steiglitz. *Combinatorial Optimization: Algorithms and Complexity*, Prentice Hall, Englewood Cliffs, NJ, 1982.
[2] M. Held and R. M. Karp. *The Traveling Salesman Problem and Minimal Spanning Trees*, Journal on Operations Research, vol. 18, pp. 1138-1162, 1970.
[3] M. M. Zavlanos and G. J. Pappas. *Dynamic Assignment in Distributed Motion Planning with Local Coordination*, IEEE Transactions on Robotics, vol. 24, no. 1, pp. 232-242, Feb. 2008.
[4] S. Kloder and S. Hutchinson. *Path Planning for Permutation-Invariant Multirobot Formations*, IEEE Transactions on Robotics, vol. 22, no. 4, pp. 650-665, Aug. 2006.
[5] M. Ji, S. Azuma, and M. Egerstedt. *Role-Assignment in Multi-Agent Coordination*, International Journal of Assistive Robotics and Mechatronics, vol. 7, no. 1, pp. 32-40, March 2006.
[6] S. L. Smith and F. Bullo. *Target Assignment for Robotic Networks: Worst Case and Stochastic Performance in Dense Environments*, In Proceedings of the 46th IEEE Conference on Decision and Control, New Orleans, LA, Dec. 2007, pp. 3585-3590.
[7] M. M. Zavlanos and G. J. Pappas. *Distributed Formation Control with Permutation Symmetries*, In Proceedings of the 46th IEEE Conference on Decision and Control, New Orleans, LA, Dec. 2007, pp. 2894-2899.
[8] N. Michael, M. M. Zavlanos, V. Kumar and G. J. Pappas. *Distributed Multi-Robot Task Assignment and Formation Control*, IEEE International Conference on Robotics and Automation, Pasadena, CA, May 2008. (to appear)
[9] H. G. Tanner, A. Jadbabaie and G. J. Pappas. *Flocking in Fixed and Switching Networks*, IEEE Transactions on Automatic Control, vol. 52, no. 5, pp. 863-868, May 2007.
[10] R. Olfati-Saber and R. M. Murray. *Consensus Problems in Networks of Agents with Switching Topology and Time-Delays*, IEEE Transactions on Automatic Control, vol. 49, no. 9, pp. 1520-1533, Sept. 2004.
[11] J. Cortes, S. Martinez and F. Bullo. *Robust Rendezvous for Mobile Autonomous Agents via Proximity Graphs in Arbitrary Dimensions*, IEEE Transactions on Automatic Control, vol. 51, no. 8, pp. 1289-1298, Aug. 2006.
[12] D. V. Dimarogonas, S. G. Loizou, K. J. Kyriakopoulos and M. M. Zavlanos. *A Feedback Stabilization and Collision Avoidance Scheme for Multiple Independent Non-point Agents*, Automatica, vol. 42, no. 2, pp. 229-243, Feb. 2006.
[13] R. Sepulchre, D. Paley, N. E. Leonard. *Stabilization of Planar Collective Motion: All-to-All Communication*, IEEE Transactions on Automatic Control, vol. 52, no. 5, pp. 811-824, May 2007.
[14] W. Ren and R. Beard. *Consensus Seeking in Multi-Agent Systems under Dynamically Changing Interaction Topologies*, IEEE Transactions on Automatic Control, vol. 50, no. 5, pp. 655-661, May 2005.
[15] T. Balch and R.C. Arkin. *Behavior-based Formation Control for Multirobot Teams*, IEEE Transactions on Robotics and Automation, vol. 14, no. 6, pp. 926-939, Dec. 1998.
[16] S. Poduri and G. S. Sukhatme. *Constrained Coverage for Mobile Sensor Networks*, In Proceedings of the IEEE International Conference on Robotics and Automation, New Orleans, LA, May 2004, pp. 165-172.
[17] M. L. Balinski. *Signature Methods for the Assignment Problem*, Journal on Operations Research, vol. 33, pp. 527-537, 1985.
[18] D. P. Bertsekas. *A New Algorithm for the Assignment Problem*, Mathematical Programming, vol. 21, pp. 152-171, 1981.
[19] U. Derigs. *The Shortest Augmenting Path Method for Solving Assignment Problems - Motivation and Computational Experience*, Annals of Operations Research, vol. 4, pp. 57-102, 1985.
[20] M. Hung. *A Polynomial Simplex Method for the Assignment Problem*, Operations Reserach, vol. 31, pp. 595-600, 1983.
[21] H. W. Kuhn. *The Hungarian Method for the Assignment Problem*, Naval Research Logistics, vol. 2, no. 1-2, pp. 83-97, March 1955.
[22] D. P. Bertsekas. *Auction Algorithms for Network Flow Problems: A Tutorial Introduction*, Computational Optimization and Applications, vol. 1, pp. 7-66, 1992.
[23] D. P. Bertsekas and D. A. Castanon. *Parallel Synchronous and Asynchronous Implementations of the Auction Algorithm*, Parallel Computing, vol. 17, pp. 707-732, 1991.
[24] G. B. Dantzig. *Linear Programming and Extenssions*, Princeton University Press, Princeton, NJ, 1963.