Asynchronous Auction for Distributed Nonlinear Resource Allocation

Ajay Kumar Bangla and David A. Castañón

Abstract-Nonlinear Resource Allocation Problems are concerned with the optimal allocation of N continuous resources to M missions/tasks with nonlinear utilities. Numerous problems from diverse fields such as search theory, statistics, finance, economics, logistics, sensor and wireless networks fit this formulation. Several centralized/synchronous algorithms are available for solving this problem including RAP Auction [1] which was proposed by the the authors. RAP Auction finds a near optimal solutions with pseudo-polynomial complexity for generalized class of monotonic convex cost functions including non-differentiable and/or non-strictly convex functions. In this paper, we present and discuss asynchronous version of RAP Auction. This algorithm supports the computation framework in which nodes behave as autonomous agents making local decisions without waiting for each other.

I. INTRODUCTION

Nonlinear resource allocation problems (RAP) are a class of optimization problems where heterogeneous resources have to be allocated to a diverse set of tasks. The underlying performance of executing a task is a nonlinear function of the bundle of resources assigned to it. Interest in RAP is motivated by diverse applications such as in search theory [2]-[4], weapon target assignment [5], [6], sensor management, market equilibria [7], production planning [8], [9], scheduling of mass screening tests [10] and allocation of software-testing resources [11]. The linear cost generalized assignment [12] and transportation [13] problems can be seen as special cases of these problems.

The history of RAPs dates back to Koopman [14], who considered distribution of a single resource to two activities, Single RAP (SRAP). Luss and Gupta [15] considered RAP as an extension of SRAP and proposed a method called Resource-Wise Optimization Algorithm (RWOA). Since then several types of algorithms have been proposed. From finite time simplex type algorithms [4], [16] to polynomial network flow based techniques [7], [17] for exponential, quadratic, or logarithmic cost functions. In [1], [18], [19], we developed a new class of pseudo-polynomial algorithm called RAP Auction. This algorithm, inspired by success of the auction algorithm for linear assignment problems, exploits the unique graph structure present in RAPs combining ideas from convex and combinatorial optimization. In essence, there is a price for each task node and in each iteration, source nodes with surpluses bid for their best tasks. The task node being bid for, decides on how much resource to accept from the bidding source node. This simple compute structure inher-

This work was supported by AFOSR grants FA9550-07-1-0361 and by ODDR&E MURI Grant FA9550-06-1-0324

The authors are with the Dept of Electrical & Computer Eng., Boston University, a jay@bu.edu, dac@bu.edu

ently makes this algorithm suitable for distributed implementation. Unlike most previous techniques, it works for all convex monotonic utilities including non-differentiable and non-strictly convex functions. All the above methods have been proposed for synchronous/centralized implementation. If implemented in distributed networks, they may require excessive information exchange and overhead.

In a distributed framework, we consider the problem as one in which a set of autonomous agents with resources collaborate to reach globally optimal allocation of their resources to a set of tasks. In spite of limited, unreliable, delayed communication or memory access capabilities and with no or minimum coordination, algorithms must converge. The foundations of asynchronous techniques and their convergence for strictly convex optimization problems are discussed in [20]. Two models for asynchronous computation, total or partial asynchronism, have been defined based on whether the delays are unbounded or bound, respectively. RAP Auction has a very simple and separable computation structure which makes it a natural candidate for distributed implementation. In this paper, we propose totally asynchronous implementations of RAP Auction for distributed RAP and prove its convergence. In this version, nodes make decisions about their allocations at arbitrary times based on outdated information.

The remainder of this paper is organized as follows. Section II formulates the RAP and briefly discusses duality. In section III, we propose asynchronous RAP Auction. Its validity is established in section IV. Section V discusses summarizes our results.

II. NONLINEAR RESOURCE ALLOCATION

Consider a bipartite graph G = (W, T, E), a triple, consisting of a set of N source nodes, a set of M sink nodes and a set of arcs, respectively. We are given, for each source $i \in W$, a scalar s_i (supply of i), for each arc $(i,j) \in E$, a positive scalar c_{ij} (gain of (i,j)) and at each sink $j \in T$ a non-increasing, closed, convex cost function $f_j: \Re^+ \mapsto \Re$. We now define the nonlinear Resource Allocation Problem (RAP) as

minimize
$$f(\mathbf{z}) := \sum_{j \in T} f_j(z_j)$$
 (1a)

minimize
$$f(\mathbf{z}) := \sum_{j \in T} f_j(z_j)$$
 (1a) subject to $\sum_{j \in T_i} x_{ij} = s_i$ $\forall i \in W$ (1b)
$$\sum_{i \in W_j} c_{ij} x_{ij} = z_j \quad \forall j \in T$$
 (1c)
$$\mathbf{x} \ge \mathbf{0}, \ \mathbf{z} \ge \mathbf{0}$$
 (1d)

$$\sum_{i \in W_j} c_{ij} x_{ij} = z_j \qquad \forall \ j \in T$$
 (1c)

$$\mathbf{x} \ge \mathbf{0}, \ \mathbf{z} \ge \mathbf{0} \tag{1d}$$

where $T_i = \{j : (i,j) \in E\}$ is the set of sinks connected to the i^{th} source, $W_j = \{i : (i,j) \in E\}$ is the set of sources connected to j^{th} sink, the real variable x_{ij} is referred to as the *flow* of the arc (i,j) and the vector $\mathbf{x} \triangleq \{x_{ij} | (i,j) \in E\}$ is referred to as the flow vector. Note z_j 's are auxiliary variables which can be viewed as *demand* for each sink and $\mathbf{z} \triangleq \{z_j | j \in T\}$. A flow vector, \mathbf{x} , is said to be *primal feasible* if demand, \mathbf{z} , is implicitly defined by (1c) and the pair (\mathbf{x}, \mathbf{z}) is primal feasible (satisfies (1b), (1c), and (1d)).

The structure of the general problem is as follows (see Fig. 1). The sink nodes can be seen as different tasks or

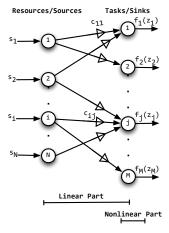


Fig. 1: Structure of Resource Allocation Problem.

missions competing for the resources of the source nodes with the objective of minimizing the overall cost. The arc gains c_{ij} model the effectiveness of assigning a particular resource to a task. In essence RAP is a nonlinear bipartite matching problem.

Introducing multipliers μ_i and p_j (also called sink *prices*) for the flow conservation constraints at the source W and sink T nodes, respectively, we get the dual of RAP as

$$\max_{\boldsymbol{\mu}, \mathbf{p}} q(\boldsymbol{\mu}, \mathbf{p})$$

where the dual function q is given by

$$q(\boldsymbol{\mu}, \mathbf{p}) = \sum_{j \in T} q_j(\boldsymbol{\mu}_{W_j}, p_j) - \boldsymbol{\mu}' \mathbf{s}$$

and q_j is defined as

$$q_{j}(\boldsymbol{\mu}_{W_{j}}, p_{j}) = \inf_{z_{j} \geq 0} \{f_{j}(z_{j}) + p_{j}z_{j}\}$$
$$+ \sum_{i \in W_{j}} \inf_{x_{ij} \geq 0} \{(\mu_{i} - c_{ij}p_{j})x_{ij}\}$$

Strong duality, existence of both primal and dual optimal solutions and existence of multipliers which satisfy *Complementary Slackness* (CS) for any primal feasible solutions were established in [1].

Following [1], we say that a flow-demand-price vector triple $(\mathbf{x}, \mathbf{z}, \mathbf{p})$ satisfy ϵ -CS, where ϵ is any positive scalar,

if $\mathbf{x} \geq 0$, $\mathbf{z} \geq 0$, $\mathbf{p} \geq 0$, and

$$c_{ij}p_j \ge \max_{k \in T_i} c_{ik}p_k - \epsilon \ \forall \ \{i, j\} \in E \text{ with } x_{ij} > 0$$
 (2)

$$-f_{i}^{+}(z_{j}) \le p_{j} \le -f_{i}^{-}(z_{j}) \ \forall \ j \in T$$
 (3)

where $f_j^-(z_j)$ and $f_j^+(z_j)$ are the left derivative and right derivative of f_j at z_j , respectively. This can be broken down in two conditions one related to the arcs and one related to the sinks:

- The flow-price pair (\mathbf{x}, \mathbf{p}) satisfies ϵ -CS_{arc} if $\mathbf{x} \ge 0$, $\mathbf{p} \ge 0$, and (2).
- The demand-price pair (\mathbf{z}, \mathbf{p}) satisfies CS_{sink} if $\mathbf{z} \ge 0$, $\mathbf{p} \ge 0$, and (3).

The triple $(\mathbf{x}, \mathbf{z}, \mathbf{p})$ satisfies ϵ -CS iff (\mathbf{x}, \mathbf{p}) and (\mathbf{z}, \mathbf{p}) satisfy ϵ -CS_{arc} and CS_{sink}, respectively. This is illustrated in Fig. 2.

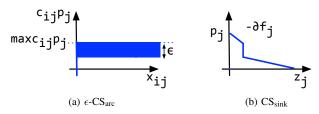


Fig. 2: Illustration of ϵ -CS. (a) All pairs of arc flows x_{ij} and $c_{ij}p_j$ should lie in the blue region, (b) All pairs of demand z_i and p_j should lie on the blue line.

The intuition behind the ϵ -CS conditions is that a feasible flow-price pair is "approximately" primal and dual optimal if the ϵ -CS conditions are satisfied as shown in this proposition which is proved in [18].

Proposition 1: Let $(\mathbf{x}^*, \mathbf{z}^*, \mathbf{p}^*)$ be a flow-demand-price triple satisfying ϵ -CS such that $(\mathbf{x}^*, \mathbf{z}^*)$ is primal feasible, then

$$0 \le f(\mathbf{z}^*) - q(\boldsymbol{\mu}^*, \mathbf{p}^*) \le \epsilon \ \mathbf{s}' \mathbf{1}$$

where μ^* is defined as $\mu_i \triangleq \max_{j \in T_i} c_{ij} p_j \quad \forall i \in W$.

III. ASYNCHRONOUS RAP AUCTION

In [1], [18], [19], we proposed a new algorithm RAP Auction. It was proposed as a synchronous algorithm as there is a strict separation between bidding and allocation phases. Delays are incurred when source nodes calculating their bids have to wait to make sure that up-to-date prices are available, and when the sinks calculating the allocations wait for all bids to come in. These penalties may be reduced by asynchronous implementation. In the asynchronous algorithm, each node can be seen as an autonomous decision maker. Sources with surpluses should be free to make bids at arbitrary times based of outdated prices and sinks allocate without waiting for all sources to submit their bids. The prices may be out-of-date because the computations of the previous phase are incomplete or communication delays. The sink nodes accept bids, update and broadcast their prices.

RAP Auction is a primal-dual method iterating simultaneously over flows and prices while preserving ϵ -CS. So

we need a map $\Phi_j: Z_j \mapsto P_j$ where $Z_j = \Re^+$ is the demand space and $P_j = \Phi_j(Z_j)$ is the price space and its inverse $\Theta_j: P_j \mapsto Z_j$ consistent with (3). For continuously differentiable and strictly convex cost functions, $\Phi_j = -\nabla f_j$ and $\Theta_j = \nabla f_j^{-1} \circ -1$ which is illustrated in Fig. 3. In [18], we show how to define these mappings without these assumptions.

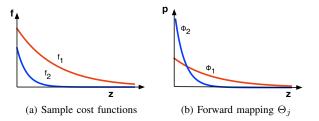


Fig. 3: Forward mapping for differentiable and strictly convex cost functions.

We use the following notation to formulate totally asynchronous RAP Auction:

- $p_j(t)$: price of sink j at time t,
- $z_i(t)$: demand at sink j at t,
- $x_{ij}(t)$: flow allocated to sink j from source i at t,
- $g_i(t)$: surplus at source i at t,

$$g_i(t) \triangleq s_i - \sum_{j \in T_i} x_{ij}(t) \quad \forall i \in W.$$

- U(t): set of all sources with positive surplus at t,
- $R(t) \subset U(t)$: set of sources with a 'ready bid' at t.

We assume that these quantities can change only at integer times t; this involves no loss of generality, since t may be viewed as the index of a sequence of physical times at which events of interest occur. Sources with positive surplus enter the set R(t) and become eligible to bid. At each time t, if R(t) is empty nothing happens. If R(t) is nonempty then a nonempty subset of R(t) bid. We assume that at time t, a source $i \in R(t)$ has used the prices $p_j(\tau_{ij}(t))$ from some earlier (but otherwise arbitrary) times $\tau_{ij}(t) \le t$ to compute its bid using the Bidding Procedure. Note $t - \tau_{ij}(t)$ is the delay in source i's copy of price of sink j at time t.

Bidding Procedure:

1) Find sink offering best value

$$j = \arg\max_{k \in T_i} c_{ik} p_k(\tau_{ij}(t)),$$

and the second best value

$$v^{\text{sec}} = \begin{cases} \max_{k \in T_i \setminus j} c_{ij} p_k(\tau_{ij}(t)) & \#T_i \ge 2\\ p_{min} & \text{else} \end{cases} . \tag{4}$$

2) Compute i's bid price for j as

$$b_i = (v^{\text{sec}} - \epsilon)/c_{ij}. \tag{5}$$

This bid price denotes the lower bound on the price of j below which flow has to be reversed (x_{ij} set to zero).

- As long as $p_j \ge b_i$, arc (i, j) satisfies ϵ -CS_{arc} i.e., sink j provides almost the best value to source i.
- 3) Compute *i*'s bid surplus $y_i = c_{ij}g_i(t)$. This is the maximum additional flow that *i* can provide to *j*.
- 4) Submit $\{y_i, b_i\}$ to sink j and set $g_i(t+1) = 0$.

The bidding procedure is illustrated in Fig. 4. A bid if accepted results in non-zero flow augmentation on arc (i, j). If at any point p_j falls below b_i , then the flow on arc (i, j) has to be reversed.

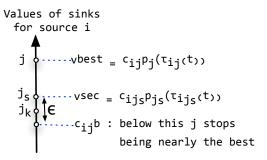


Fig. 4: Illustration of the bidding phase involving a source i. Sinks j and j_s that offer the best and second best value, respectively, are determined. Source i bids for j with bid price b.

When a sink receives a new bid $\{y_i^{\text{new}}, b_i^{\text{new}}\}$, it is possible that $b_i^{\text{new}} > p_j(t)$ as b_i^{new} was computed using outdated prices and such bids have to be obviously rejected. However only rejecting bids above p(t) is not sufficient for finite termination. Bids should only be accepted if $p_j(t) \geq b_i^{\text{new}} + \epsilon/c_{ij}$. We call such bids *substantive*. This ensures that between flow augmentation and reversal on a arc (i,j), the price of j has to drop by at least ϵ/c_{ij} . This property will be crucial in the proof for finite time termination. For ease of exposition on how substantive bids are allocated, we use the following representation for previously accepted and still valid bids at a given sink j:

• $B_j = \{b_{i_1}, b_{i_2}, \dots, b_{i_n}\}$: List of accepted bid prices in order of decreasing value, i.e.

$$p_i(t) \ge b_{i_1} \ge b_{i_2} \ge \ldots \ge b_{i_n}$$

- $Y_j = \{y_{i_1}, y_{i_2}, \dots, y_{i_n}\}$: List of flows received where y_{i_k} is an alias for $c_{i_k j} x_{i_k j}(t)$ with b_{i_k} as the corresponding bid price.
- We use b_{i_0} as an alias for $p_j(t)$.

A source i_k 's bid $\{y_{i_k}, b_{i_k}\}$ is valid if $p_j \geq b_{i_k}$ and $y_{i_k} > 0$. The state at the sink j at the beginning of the allocation phase is illustrated in Fig. 5. As flow starts getting accepted from i during the current iteration, the demand z_j increases and the demand-price pair slides to the right of 'A' along the blue curve. We call this mode of acceptance as *absorption*. Up to δ_0 of flow can be absorbed before reaching 'B'. This is called as the *demand margin* between successive prices p_j and b_{i_1} . In general, we define demand margin, δ_k , as

$$\delta_k := \Theta_i(b_{i_{k+1}}) - \Theta_i(b_{i_k})$$

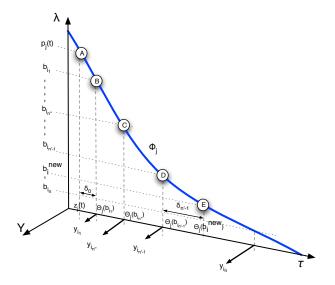


Fig. 5: State of sink j at the beginning of allocation phase. 'A' corresponds the current demand-price pair $(z_j(t), p_j(t))$ at the beginning of the allocation procedure. b_i^{new} and $\{b_{i_1}, \ldots, b_{i_n}\}$ correspond to the current and old bid prices, respectively, with corresponding flows $\{y_{i_1}, \ldots, y_{i_n}\}.$

If $y_i^{\text{new}} > \delta_0$, then the demand-price pair will reach 'B'. At 'B', flow can't be absorbed as before since the source i_1 's bid will become invalid if p_i falls below b_{i_1} . So while at 'B' flow from i is accepted by reversing flow to i_1 . We call this mode of flow acceptance as reversal. The price and demand don't change during this mode. If $y_i^{\text{new}} \geq \delta_0 + y_{i_1}$, then this bid is completely reversed. Now between b_{i_1} and b_{i_2} upto δ_1 can be absorbed before reversing i_2 's bid and so forth till either p_i drops to b_i^{new} or all of y_i^{new} is accepted (exhausting push). This logic is carried out in the allocation procedure which is executed when a sink j receives a bid $\{y_i^{\text{new}}, b_i^{\text{new}}\}$.

Allocation Procedure

- 1) If $p_i(t) \geq b_i^{\text{new}} + \epsilon/c_{ij}$
 - a) If sink j has previously accepted bid from i
 - i) Update $b_i = b_i^{\text{new}}$ and re-sort B_j and Y_j , accord-
 - b) Else insert b_i^{new} and 0 in B_j and Y_j , respectively, in sorted order and let n' be the index such that $i'_n = i$.
 - c) If $y_i^{\text{new}} \ge y_{\text{max}}$ where

$$y_{max} = \sum_{k=1}^{n'-1} y_{i_k} + \Phi_j(b_i^{\text{new}}) - z_j(t).$$

Non-exhausting push:

- i) Reverse bids from sources $\{i_1, \ldots, i_{n'-1}\}$ while accepting y_{max} from source i,
- ii) Set $p_j(t+1)=b_i^{\mathrm{new}}$ and $z_j(t+1)=\sum_{k=n'}^n y_{i_k}.$ iii) Set unaccepted surplus $y=y_i^{\mathrm{new}}-y_{max}.$
- d) Else

Exhausting push: Determine the set of sources $\{i_k:$

 $1 \le k \le n''$ whose bids have to be completely reversed where n'' < n'. Source i_k 's bid has to be reversed if k < n' and

$$\sum_{l=1}^{k} (y_{i_l} + \delta_{l-1}) \le y_i^{\text{new}}.$$

- i) If $y_i^{\text{new}} \leq \sum_{k=1}^{n''} (y_{i_k} + \delta_{k-1}) + \delta_{n''}$ Allocation with complete reversals and absorp-
 - A) Reverse bids from $\{i_1, \ldots, i_{n''}\}$ completely
 - while accepting y_i^{new} from i, B) Set $z_j(t+1) = \sum_{k=n''+1}^n y_{i_k}$ and $p_j(t+1) = \Phi_j(z_j(t+1))$.
- ii) else

Allocation with at least one partial reversal:

- A) Reverse $\{i_1, \ldots, i_{n''}\}$ bids completely and
- $b_{i_{n''+1}}$ partially while accepting y_i^{new} from i, B) Set $z_j(t+1)=\sum_{k=n''+1}^n y_{i_k}$ and $p_j(t+1)=$
- iii) Set unaccepted surplus y = 0.
- 2) Else set unaccepted surplus $y = y_i^{\text{new}}$.
- 3) If y > 0, return unaccepted surplus y to i.

Reversing bid to source i_k implies setting $x_{kj}(t+1) = 0$, returning surplus y_{i_k} to i_k and deleting b_{i_k} and y_{i_k} from B_j and Y_i , respectively. Whenever a source i receives a surplus y back from sink j then it updates its surplus as

$$g_i(t+1) = g_i(t) + y/c_{ij}.$$

The algorithm can be started from any (x, z, p) satisfying ϵ -CS, (1c) and $g_i \geq 0 \ \forall \ i \in W$. As an initial choice, we may use

$$\mathbf{z}(1) = \mathbf{0}, \ \mathbf{x}(1) = \mathbf{0}, \ \text{and}$$

$$p_j(1) = \Phi_j(0) \qquad \forall \ j \in T.$$

IV. VALIDITY AND CONVERGENCE

To maximize the generality and flexibility of our model, the precise mechanism by which prices and surplus return communications are done is left unspecified subject to the following two assumptions:

Assumption 1: U(t): nonempty $\Rightarrow R(t')$: nonempty for some t' > t.

Assumption 2: For all i, j, and t,

$$\lim_{t \to \infty} \tau_{ij}(t) = \infty.$$

 $\lim_{t\to\infty}\tau_{ij}(t)=\infty.$ Clearly an asynchronous algorithm cannot solve the problem if sources with surplus stop submitting bids, sinks do not process submitted bids and if old information is not eventually discarded. This is the motivation for the preceding two assumptions.

The following propositions establishes the validity of the asynchronous RAP Auction algorithm.

Proposition 2: The algorithm preserves ϵ -CS, (1c), and non-negative source surpluses throughout its execution; that is, if the allocation and prices at the beginning of the iteration satisfy ϵ -CS, (1c), and $\mathbf{g} \ge 0$, then same is true at the end of iteration.

Proof: Since flows are rejected or reversed before sink prices fall below the bid prices, we have

$$x_{ij}(t) > 0 \Rightarrow p_j(t) \ge b_i$$

where b_i was computed using old prices. Observe that $\{\mathbf{z}(t)\}$ is a non-decreasing sequence. It increases strictly during absorption and is unchanged during reversals. From convexity and monotonicity, it follows that $\{\mathbf{p}(t)\}$ is a non-increasing sequence. From this monotonicity of prices, (5) and (4), we have

$$c_{ij}b_i \ge \begin{cases} \max_{k \in T_i \setminus j} c_{ik} p_k(t) - \epsilon & \#T_i \ge 2 \\ p_{min} - \epsilon & \text{else} \end{cases}.$$

If $p_j(t) \geq b_i$, then

$$c_{ij}p_j(t) \ge \max_{k \in T_i} c_{ik}p_k(t) - \epsilon.$$

Hence ϵ -CS_{arc} is preserved on all arcs with positive flows. Steps 1(c)ii, 1(d)iB and 1(d)iiB in Allocation Procedure enforce (1c) and CS_{sink}. So for all the arcs and sinks, we have ϵ -CS_{arc} and CS_{sink}, respectively. Hence ϵ -CS holds for triple ($\mathbf{x}(t), \mathbf{z}(t), \mathbf{p}(t)$).

Surpluses of bidding sources can decrease at most to zero. The surpluses at other sources can only increase due to reversals of their flows. So all the surpluses remain nonnegative.

The algorithm can be stopped as soon as the set of sources with surplus U(t) is empty. We say that the algorithm terminates at time t if t is the first time U(t) is empty. It follows from the above proposition that if the algorithm terminates, it does so with an ϵ -CS satisfying flow-demand-price triple $(\mathbf{x}^*, \mathbf{z}^*, \mathbf{p}^*)$ such that $(\mathbf{x}^*, \mathbf{z}^*)$ is primal feasible as the termination criterion $\mathbf{g} = \mathbf{0}$ enforces (1b). So the validity of asynchronous RAP Auction rests on whether it terminates finitely.

Proposition 3: The asynchronous RAP Auction terminates in finite time.

Proof: We note the following:

1) By definition (5)

$$b_i < p_i \left(\tau_{ii}(t) \right) - \epsilon / c_{ii}. \tag{6}$$

2) Suppose a sink receives an infinite number of bids during the algorithm. Then, an infinite subset of these bids must be substantive; otherwise $p_j(t)$ would stay constant for t sufficiently large, we would have $p_j(\tau_{ij}(t)) = p_j(t) = p_j$ for t sufficiently large because old price information is eventually purged from the system (cf. Assumption 2), and in view of (6), we would have $b_i \leq p_j(t) - \epsilon/c_{ij}$ for all times t at which j receives a bid, arriving at a contradiction.

Assume now, in order to obtain a contradiction, that the algorithm does not terminate finitely. Then, because of Assumption 1, there is an infinite number of times t at which R(t) is nonempty and at each of these times, at least one sink receives a bid. Thus, there is a nonempty subset of sink which

receives an infinite number of bids. In view of (2), a subset of these sinks receive infinite number of substantive bids. From lemma 1, price of at least one sink drop infinitely often by at least ϵ_{min} . But this is a contradiction as the sequence $p_j(t)$ is a non increasing sequence upper and lower bounded by $\Phi_j(0) < \infty$ and $\Phi_j(\sum_{i \in W_i} c_{ij} s_i) > 0$, respectively.

Lemma 1: After processing a finite number of substantive bids, prices drop by at least ϵ_{min} or the algorithm terminates.

Proof: Assume the contrary that there is a sequence of infinite successive iterations, denoted by $\Delta = \{t_1, t_2, \dots\}$, such that

 $\left\|\mathbf{p}(1) - \lim_{t \to \infty} \mathbf{p}(t)\right\|_{\infty} < \epsilon_{min}.$

Define n(t) = #U(t), number of source nodes with nonzero surplus at time t and its variation as

$$\nabla n(t) = n(t+1) - n(t). \tag{7}$$

Without loss of generality, we assume that only one substantive bid is processed per time. Since bid processing across sinks is completely decoupled and sequential within sinks, this assumption can be easily relaxed by subdividing each time slot according to the number of substantive bids processed. During this interval every substantive bid results in a exhausting push and the bidding source is exhausted. Based on how the flow is accepted at a sink node, an iteration can be classified as:

1) Reversal Push: In such iterations, flow is entirely or partially accepted by reversal. Such iterations can decrease or increase n(t+1), i.e.,

$$\nabla n(t) \ge -1$$
.

Flows allocated during Δ , being substantive can't be reversed unless the price drops by at least ϵ_{min} . So at each sink j, there are only finite number of arcs, say m_j , each with finite flow which can be reserved i.e.,

$$p_j(t_0) \ge b_{i_1} \ge \cdots \ge b_{i_{m_j}} > p_j(t_0) - \epsilon_{min}$$

where

$$\{b_{i_1},\ldots,b_{i_{m_j}}\}\subset B_j.$$

We denote this set of reversible arcs as

$$E_{\text{rev}} = \{(i, j) \in E : x_{i,j}(t_0) > 0 \& b_{ij} > p_j(t_0) - \epsilon_{min} \}$$

where $t_0 = t_1 - 1$. For any arc in E_{rev} , $\{x_{i,j}(t)\}$ is a decreasing sequence. If the flow becomes zero, we say that the arc has *saturated*.

We now argue that there can only be finite number of reversal iterations. Assume that there can be infinite iterations with reversals. Then there at least one source i which makes infinite substantive exhausting bids for at least one sink j and some arc (\hat{i}, j) where flow is strictly reversed infinite number of times. To be able to make infinite exhausting bids, there has to be some sink other than j which reverses flow to i. Continuing in this manner, we can construct sequence of infinite circulations in at least one cycle containing forward arc (i, j) and reverse arc (\hat{i}, j) . The gain of any such cycle

where each arc satisfies ϵ -CS_{arc} is greater than unity [18]. So after a finite circulations in this cycle, at least one of the reverse arcs gets saturated. Hence we arrive at a contradiction.

 Non Reversal Push (NRP): In such iterations, flow is completely accepted without any reversal which strictly decreases the number of sources with surplus, i.e.,

$$\nabla n(t) = -1 \quad \forall \ t \in \Delta_{NRP}. \tag{8}$$

where Δ_{NRP} is the subsequence of Δ with NRP iterations. We show that there can only be finitely many such iterations by showing that n(t) can't increase arbitrarily. The subsequence of Δ for which n(t) strictly increases is defined as

$$\Delta^+(t) \triangleq \{t_1 \le l < t : \nabla n(l) > 0\}. \tag{9}$$

This happens if at least two flow reversals (one complete and one at least partial) take place during a given iteration as the bidding source is itself exhausted. Since there are only finite reversible arcs, we have for all $t \geq t_1$

$$\#\Delta_+(t) \leq \#E_{\text{rev}}$$

and the positive variation is also bounded as an arc once saturated remains saturated.

$$\sum_{l \in \Delta^{+}(t)} \nabla n(l) \le \# E_{\text{rev}}.$$
 (10)

For any $t \ge t_1$, from (7)

$$n(t) - n(t_1) = \sum_{l \in t_1}^{t-1} \nabla n(l)$$

$$= \sum_{l \in \Delta^+(t)} \nabla n(l) + \sum_{l \in \Delta_{NRP}(t)} \nabla n(l)$$

$$\leq \# E_{rev} - \# \Delta_{NRP}(t)$$

where $\Delta_{NRP}(t) = \{l \in \Delta_{NRP} : l \leq t\}$. The last inequality follows from (10), (8) and

$$\nabla n(l) = 0 \quad \forall \ l \notin \Delta^+(t) \bigcup \Delta_{NRP}(t).$$

So

$$\#\Delta_{NRP} \le \#E_{rev} - n(t) + n(t_0) \le M(N-1) + N.$$

Hence we can only have finitely many iterations without the prices dropping by at least ϵ_{min} .

V. Conclusion

We have proposed and analyzed totally asynchronous version of RAP Auction for solving RAPs in a distributed fashion. This technique views the nodes as autonomous agents which collaborate with each other to reach a near optimal allocation.

REFERENCES

- [1] A. K. Bangla and D. A. Castañón, "Auction algorithm for nonlinear resource allocation problems," *Proc.* 49th IEEE CDC, Dec. 2010.
- [2] B. O. Koopman, "The theory of search. iii. the optimum distribution of searching effort," *Oper. Res.*, vol. 5, no. 5, Oct 1957.
- [3] A. Charnes and W. W. Cooper, "The theory of search: Optimal distribution of effort," Mgmt. Sci., vol. 5, 1958.
- [4] A. R. Washburn, "Finite method for a nonlinear allocation problem," J. Optm. Theory Appl., vol. 85, no. 3, pp. 705–726, June 1995.
- [5] —, "Sortie optimization and munitions planning," *Military Oper. Res.*, pp. 13–18, 1994.
- [6] R. K. Ahuja, A. Kumar, K. C. Jha, and J. B. Orlin, "Exact and heuristic algorithms for the weapon-target assignment problem," *Oper. Res.*, vol. 55, no. 6, pp. 1136–1146, Nov 2007.
- [7] N. R. Devanur, C. H. Papadimitriou, A. Saberi, and V. V. Vazirani, "Market equilibrium via a primal–dual algorithm for a convex program," *J. ACM*, vol. 55, no. 5, pp. 1–18, 2008.
- [8] T. Ibaraki and N. Katoh, Resource Allocation Problems: Algorithmic approaches. The MIT Press, 1988.
- [9] M. Patriksson, "A survey on the continuous nonlinear resource allocation problem," *Eur. J. Oper. Res.*, vol. 185, no. 1, pp. 1–46, Feb. 2008
- [10] H. L. Lee and W. P. Pierskalla, "Mass screening models for contagious diseases with no latent period," *Oper. Res.*, vol. 36, no. 6, pp. 917–928, 1988.
- [11] H. Ohtera and S. Yamada, "Optimal allocation and control problems for software-testing resources," *IEEE Trans. Reliability*, vol. 39, no. 2, pp. 171–176, Jun 1990.
- [12] D. P. Bertsekas, Network Optimization: Continuous and Discrete Models. Athena Scientific, 1998.
- [13] Y. Censor and S. A. Zenios, Parallel Optimization: Theory, Algorithms and Applications. Oxford University Presss, 1997.
- [14] B. O. Koopman, "The optimum distribution of effort," Open. Res., vol. 1, no. 2, Feb 1953.
- [15] H. Luss and S. . K. Gupta, "Allocation of effort resources among competing activities," Oper. Res., 1975.
- [16] J. M. Einbu, "A finite method for the solution of a multi-resource allocation problem with concave return functions," *Mathematics of Oper. Res.*, vol. 9, 1984.
- [17] K. M. Mjelde, "The allocation of linear resources to concave activitiesa finite algorithm with a polynomial time bounds," J. Oper. Res. Society, vol. 33, 1982.
- [18] A. K. Bangla and D. A. Castañón, "RAP Auction: Auction algorithm for nonlinear resource allocation problems," 2010, CISE Report, Boston University.
- [19] —, "Pseudo polynomial auction algorithm for nonlinear resource allocation problems," to appear in IEEE ACC, June. 2011.
- [20] D. P. Bertsekas and J. N. Tsitsiklis, Parallel and Distributed Computation: Numerical Methods. Athena Scientific, 1997.