2011 American Control Conference
on O'Farrell Street, San Francisco, CA, USA
June 29 - July 01, 2011

Pseudo-Polynomial Auction Algorithm for Nonlinear Resource Allocation

Ajay Kumar Bangla and David A. Castañón

***Abstract*— We study the problem of optimally assigning N divisible resources to M competing tasks. This is called the Nonlinear Resource Allocation Problem (RAP). Recently, we proposed a new algorithm, RAP Auction [1], which finds a near optimal solution in finite time. It works for monotonic convex cost functions and has a simple computation structure. In this paper, we propose an elegant extension which enables RAP Auction to have pseudo-polynomial complexity.**

I. INTRODUCTION

Let us consider the problems where heterogeneous resources have to be allocated in continuous amounts 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. Problems of this type arise in diverse applications such as search theory [2]–[4], weapon target assignment [5], [6], sensor management, and market equilibria [7] .

In [1], [8], we developed a new class of finite time algorithms called RAP Auction for solving such problems. This algorithm was inspired by success of the auction algorithm for linear assignment problems. 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 inherently makes this algorithm suitable for distributed implementation. It works for generalized monotonic convex functions including non-differential or/and non-strictly convex functions.

RAPs are convex optimization problems on generalized network (network with gains). Such network are usually harder than their ordinary network counterparts because cycles in generalized networks can generate or absorb flow. It is the presence of such cycles that prevents RAP Auction from having stronger complexity results than finite termination. One resolution to this dilemma is to detect and resolve cycles as proposed in [9]. However, this requires additional bookkeeping and imposes a certain sequential bidding order.

In this paper, we propose a simple yet potent extension to RAP Auction which enables it to have pseudo polynomial complexity. This is achieved by addition of a single new step without any need for cycle detection or additional bookkeeping or imposing a bidding order. Thus it preserves all the benefits of the existing computation structure. The convergence proof for this simple extension is novel and non-trivial.

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, `ajay@bu.edu, dac@bu.edu`

The remainder of this paper is organized as follows. In section II, we formulate the RAP and briefly discuss duality for RAP. Section III reviews RAP Auction algorithm. In section IV, we propose our extensions to RAP Auction and prove its convergence. Some numerical experience is reported in section V. Section VI summarizes our results.

II. PROBLEM FORMULATION

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

$$\min_{\mathbf{x},\mathbf{z}} \quad f(\mathbf{z}) := \sum_{j \in T} f_j(z_j) \tag{1a}$$

$$\text{subject to} \quad \sum_{j \in T_i} x_{ij} = s_i \qquad \forall\, i \in W \tag{1b}$$

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

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

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, $\mathbf{x} \triangleq \{x_{ij} | (i, j) \in E\}$ is the flow vector, and $\mathbf{z} \triangleq \{z_j | j \in T\}$ is the *demand* vector.

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 [8].

Following [1], we say for any positive scalar ϵ, a triple $(\mathbf{x}, \mathbf{z}, \mathbf{p})$ is ϵ-CS iff $(\mathbf{x}, \mathbf{p})$ and $(\mathbf{z}, \mathbf{p})$ are ϵ-CS$_{\text{arc}}$ and CS$_{\text{sink}}$, respectively, where

- flow-price pair $(\mathbf{x}, \mathbf{p})$ is ϵ-CS$_{\text{arc}}$ if $\mathbf{x} \geq 0$, $\mathbf{p} \geq 0$, &
$$c_{ij}p_j \geq \max_{k \in T_i} c_{ik}p_k - \epsilon \;\forall\; \{i, j\} \in E \text{ with } x_{ij} > 0,$$
- demand-price pair $(\mathbf{z}, \mathbf{p})$ is CS$_{\text{sink}}$ if $\mathbf{z} \geq 0$, $\mathbf{p} \geq 0$, &
$$-f_j^+(z_j) \leq p_j \leq -f_j^-(z_j) \;\forall\; j \in T$$
where $f_j^-(z_j)$ and $f_j^+(z_j)$ are the left and right derivatives of f_j, respectively.

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 was proved in [8].

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
$$0 \leq f(\mathbf{z}^*) - q(\boldsymbol{\mu}^*, \mathbf{p}^*) \leq \epsilon \; \mathbf{s}'\mathbf{1}$$
where $\boldsymbol{\mu}^*$ is defined as $\mu_i \triangleq \max_{j \in T_i} c_{ij}p_j \quad \forall\; i \in W$.

III. RAP Auction Algorithm

RAP Auction is a primal-dual method. In a typical iteration, prices have to be computed for given demands, and demands for given prices which are CS$_{\text{sink}}$ consistent. 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 $\Theta_j : P_j \mapsto Z_j$. We use the following notation to describe the RAP Auction Algorithm:

- $p_{min} \triangleq \min_{j \in T} \inf P_j \geq 0$,
- $\epsilon_{min} \triangleq \epsilon / \max_{ij \in E} c_{ij}$,
- $\mathbf{p}(t), \mathbf{x}(t), \mathbf{z}(t)$ denote the price, flow and demand vectors at the beginning of t^{th} iteration, respectively,
- g_i denotes the surplus of source $i \in W$:
$$g_i(t) \triangleq s_i - \sum_{j \in T_i} x_{ij}(t) \qquad \forall\; i \in W.$$

At the start of the generic t^{th} iteration we have flow-demand-price vector triple $(\mathbf{x}(t), \mathbf{z}(t), \mathbf{p}(t))$ that satisfies ϵ-CS, the flow-demand pair $(\mathbf{x}(t), \mathbf{z}(t))$ satisfies (1c) and $\mathbf{g}(t) \geq 0$. A generic iteration consists of two phases: the bidding phase and the allocation phase, which we now describe.

Bidding Phase:

Select a source $i \in W$ with $g_i(t) > 0$; if no such source can be found, the algorithm terminates.

1) Compute the current value $v_{ij} = c_{ij}p_j(t)$ of each sink $j \in T_i$. Find a sink j_1 offering the best value
$$v_{\text{best}} = \max_{j \in T_i} v_{ij},$$
and a sink j_2 offering second best value
$$v_{\text{sec}} = \max_{j \in T_i \setminus j_1} v_{ij}.$$
2) Compute i's bid price for j_1 as
$$b = \begin{cases} (v_{\text{sec}} - \epsilon)/c_{ij_1} & \#T_i \geq 2 \\ p_{min} & \text{else} \end{cases}. \tag{2}$$
3) Compute i's bid surplus $y = c_{ij_1}g_i$.
4) Submit the bid $\{y, b\}$ to sink j_1.

When a sink, say j, receives a bid, it runs the allocation phase. For ease of exposition, we use the following representation for previously accepted and still valid bids at a given sink j:

- $B_j = \{b_{i_1}, \ldots, b_{i_n}\}$: List of accepted bid prices in decreasing order,
- $Y_j = \{y_{i_1}, \ldots, y_{i_n}\}$: 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 $p_j(t) \geq b_{i_k}$ and $y_{i_k} > 0$. The state at sink j at the beginning of the allocation phase

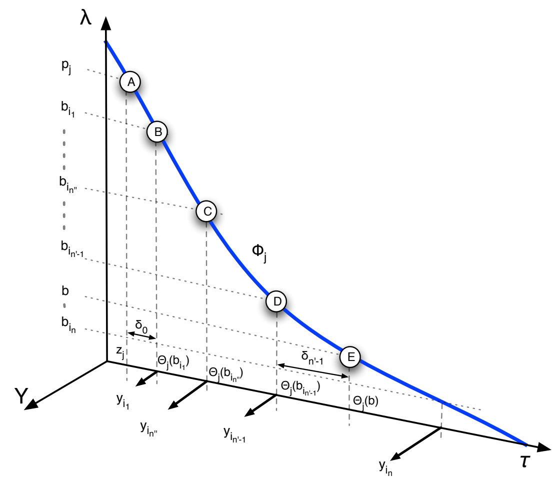


Fig. 1: State of sink j at the beginning of allocation phase. 'A' corresponds the current demand-price pair $(z_j(t), p_j(t))$ at the beginning of the allocation phase. b and $\{b_{i_1}, \ldots, b_{i_n}\}$ correspond to the current and old bid prices, respectively, with corresponding flows $\{y_{i_1}, \ldots, y_{i_n}\}$.

is illustrated in Fig. 1. 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 the *demand margin* between successive prices $p_j(t)$ and b_{i_1}. In general, we define demand margin, δ_k, as
$$\delta_k := \Theta_j(b_{i_{k+1}}) - \Theta_j(b_{i_k}).$$

If $y > \delta_0$, then the demand-price pair will reach 'B'. 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 $y \geq \delta_0 + y_{i_1}$, then there is completely reversal to i_1. In this case, we say reverse arc (j, i_1) has *saturated*. 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_j drops to b or surplus at source i is exhausted (*exhausting push*). This logic is carried out during the allocation phase.

Allocation Phase:

Select a sink $j \in T$ which received a new bid $\{y, b\}$.

1) If sink j has previously accepted bid from i
   a) Update $b_i = b$ and re-sort B_j and Y_j, accordingly.
2) Else insert b and 0 in B_j and Y_j, respectively, in sorted order and let n' be the index such that $i'_n = i$.
3) If $y \geq y_{\max}$ where $y_{max} = \sum_{k=1}^{n'-1} y_{i_k} + \Phi_j(b) - z_j(t)$
   Non-exhausting push:
   a) Reverse bids from $\{i_1, \ldots, i_{n'-1}\}$ while accepting y_{max} from i,
   b) Set $p_j(t+1) = b$ and $z_j(t+1) = \sum_{k=n'}^{n} y_{i_k}$.
4) Else
   Exhausting push: Determine sources $\{i_k : 1 \leq k \leq n''\}$ whose bids will be completely reversed where $n'' < n'$. i_k's bid is reversed if $k < n'$ and $\sum_{l=1}^{k}(y_{i_l} + \delta_{l-1}) \leq y$.
   a) If $y \leq \sum_{k=1}^{n''}(y_{i_k} + \delta_{k-1}) + \delta_{n''}$
      Allocation with complete reversals and absorptions:
      i) Reverse bids from $\{i_1, \ldots, i_{n''}\}$ completely while accepting y from i,
      ii) Set $z_j(t+1) = \sum_{k=n''+1}^{n} y_{i_k}$ and $p_j(t+1) = \Phi_j(z_j(t+1))$.
   b) else
      Allocation with at least one partial reversal:
      i) Reverse $\{i_1, \ldots, i_{n''}\}$ bids completely and $b_{i_{n''+1}}$ partially while accepting y from i,
      ii) Set $z_j(t+1) = \sum_{k=n''+1}^{n} y_{i_k}$ and $p_j(t+1) = b_{i_{n''+1}}$.

(Reversing bid to i_k implies $x_{kj}(t+1) = 0$ and deleting b_{i_k} and y_{i_k} from B_j and Y_j, respectively.)

We showed that RAP terminates in a finite number of iteration with a near optimal solution.

Proposition 2: The RAP auction terminates after finite iterations with an ϵ-CS satisfying flow-demand-price triple $(\mathbf{x}^*, \mathbf{z}^*, \mathbf{p}^*)$ such that $(\mathbf{x}^*, \mathbf{z}^*)$ is primal feasible.

This was derived by first establishing the following lemmas:

Lemma 1: Every iteration preserves ϵ-CS, (1c), & $\mathbf{g} \geq 0$.

Lemma 2: After finite number of iteration, price drops by at least ϵ_{min}.

IV. Revised RAP Auction

In this section, we illustrate why RAP Auction can't have stronger complexity bounds and then propose our extension. First we define necessary terminology. At any iteration, we can define the set A that contains arcs oriented in the direction of flow change. In particular, for each source $i \in W$, A contains one forward arc $(i, j) \in E$ such that if i were to bid, it would bid for j, i.e.,

$$j = \arg\max_{k \in T_i} c_{ik} p_k,$$

and for each sink $j \in T$, A contains a backward arc for each valid bid leveled with p_j, i.e., if $p_j = b_{ij}$, then $(j, i) \subset A$.

Now (W, T, A) defines the *admissible graph* G. This G can be cyclic. Due to ϵ-CS, all such cycles are flow generating, i.e, cycle gain ≥ 1 [8]. As illustrated in Fig. 2, the complexity

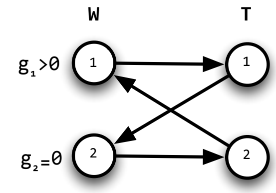


Fig. 2: Impact of cyclic admissible graph. Assume at t we have this graph. In this iteration source 1 bids for sink 1 increasing x_{11} by g_1 while reducing x_{21} by $g_1 c_{11}/c_{21}$. Next 2 bids for 2 increasing x_{22} by $g_1 c_{11}/c_{21}$ and reducing x_{12} by $g_1\gamma$ where $\gamma = c_{22}c_{11}/c_{21}c_{12}$, the cycle gain. After k such circulations x_{21} and x_{12} are reduced by $g_1\gamma^{k-1}c_{11}/c_{21}$ and $g_1\gamma^k$, respectively. This sequence continues till one of the reverse arcs saturates making the graph acyclic. Since $\gamma \geq 1$, this happens in order $O(1/g_1)$ iterations.

of resolving such cycles though finite can be arbitrarily large. This prevents RAP Auction from having stronger than finite termination.

We propose appending this step to the bidding phase which resolves such cycles.

Bidding Phase (contd.):

5) Update all the valid bids of i

$$b_i = (v_{\text{best}} - \epsilon)/c_{ij} \ \forall \ j \in T_i \backslash j_1 : b_i \in B_j. \tag{3}$$

The bid prices determine the sink prices at which flows have to be reversed so as not to violate CS$_{\text{sink}}$. In this new step, the bidding source revises the bid prices for all its valid bids, excluding the current bid as shown in Fig. 3. If its current best value, v_{best}, has strictly reduced, then all its valid bid prices are strictly lowered.

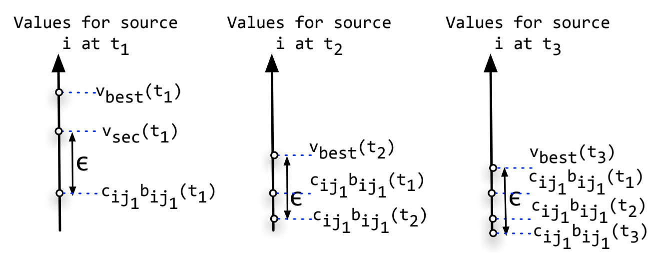


Fig. 3: Illustration of bid update step. Let $t_1 < t_2 < t_3$ be iterations in which i bids subsequently. In t_1, i bids for j_1 setting b_{ij_1} using (2). At t_2, if i bids for some sink other than j_1 and $v_{sec}(t_1) > v_{best}(t_2)$, then b_{ij_1} is strictly lowered in the bid update step. This step again revises b_{ij_1} at t_3 if $v_{\text{best}}(t_3) < v_{\text{best}}(t_2)$.

Trivially lemma 1 continues to hold with this modification. For the convergence result, we first have to re-derive lemma 2 under the new setting.

Lemma 3: The number iterations between two successive price drops is bounded by $O(N^2)$.

Proof: An iteration in which the prices don't change is called *Non Price Reducing* (NPR). Every NPR iteration is exhausting and flow is completely accepted by reversal. Let $\{1, 2, \dots, \chi\}$ denote a sequence of successive iterations between two non zero price drops. In this sequence, let $\dot{W} = \{i_1, \dots, i_\eta\}$ and $\dot{T} = \{j_1, \dots, j_\eta\}$ denote the sources and sinks making and receiving bids, respectively such that i_1 bids for j_1, i_2 bids for j_2. Since more than one sources can bid for the same sink, $\dot{T}$ may not contain all unique sinks. Let $\dot{G} = (\dot{W}, \dot{T}, \dot{A})$ denote the corresponding admissible graph. If this graph is acyclic, then from Lemma 4, the number of NPR iterations is bounded by

$$\chi_{\text{acyclic}}(\eta) \leq \eta(\eta + 1)/2.$$

However if there are cycles, for RAP Auction we only have $\chi < \infty$. With the new proposed bid update step, we obtain a polynomial bound as we now show.

For each sink $j \in \dot{T}$, let $t_j < 1$ denote the latest price reducing iteration, i.e., $p_j(t_j) - p_j(t_j + 1) > 0$. Assume that set of sinks $\dot{T}$ is labeled such that $t_{j_1} \leq t_{j_2} \leq \cdots \leq t_{j_\eta}$. For each sink j, we define its *push list* as set of following sources whose flows can be reversed

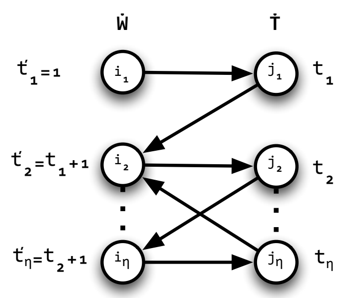


Fig. 4: Illustration of the admissible graph during a series of NPR iterations. The arcs specify the direction along which flow can changed according to the rules of the algorithms. The push lists are $L_{j_1} = \{i_2\}, L_{j_2} = \{i_\eta\}, \dots, L_{j_\eta} = \{i_2\}$.

$$L_j = \{i \in \dot{W} | p_j = b_{ij}\}.$$

Now consider one such bid b_{ij} where $i \in L_j$. Then this bid price hasn't changed since $t_j + 1$. Due to the new bid update step, this is either because the source i hasn't bid again or its best value hasn't changed since $t_j + 1$. So for each source $i \in \dot{W}$, we can define

$$t'_i = \begin{cases} 0 & i \notin L_j \ \forall j \in \dot{T} \\ \min_{\{j \in \dot{T} | i \in L_j\}} t_j & \text{else} \end{cases}.$$

Then all the bid prices of i haven't changed since $t'_i + 1$. If i were now to bid for a sink j whose price has changed after $t'_i + 1$ ($t'_i \leq t_j$), it means that the best value for i has strictly reduced. Accordingly, in the new step 5, the bid prices are strictly lowered and the corresponding reverse arcs are removed from $\dot{G}$. For example in Fig. 4, i_2's bid prices for its allocations to j_1 and j_η hasn't changed since $t_1 + 1$ and when it bids again, these bid prices are strictly lowered removing (j_1, i_2) and (j_η, i_2) from $\dot{G}$. Such a source can't bid again in this sequence as it is exhausted in the current iteration and its flow can't be reversed without a drop in prices. We call such a source as *Single Push Source* (SPS).

Fig. 5 is the biadjacency matrix of Fig. 4. Because of the ordering of the sink nodes, all the SPSs correspond to rows with negative lower diagonal elements. From this, it is easy to make the following observations:

	t_1 j_1	t_2 j_2	$\dots$ $\dots$	t_η j_η
i_1	1	0	$\dots$	0
i_2	-1	1	$\dots$	-1
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
i_η	0	-1	$\dots$	1

Fig. 5: Biadjacency matrix for Fig. 4. The diagonal $+1$ elements correspond to the forward arcs and -1 off-diagonal elements correspond to the reverse arcs. Negative lower diagonal elements correspond to SPSs. Here i_2 and i_η are SPSs.

1) $\dot{G}$ is acyclic if it has no SPSs.
2) Every cycle in $\dot{G}$ has at least one SPS.

When an SPS bids, then all the non-diagonal entries in its row are set to zero. So there can be at most one circulation in any cycle. Without the SPSs participating, the admission graph is essentially acyclic and after all the SPSs have bid once, $\dot{G}$ becomes acyclic, i.e.

$$\chi \leq \chi_{\text{acyclic}}(\eta - \eta_{\text{SPS}}) + \eta_{\text{SPS}} + \chi_{\text{acyclic}}(\eta - \eta_{\text{SPS}}) \leq (\eta + 1)^2$$

where η_{SPS} is the number of SPSs. So finally we have χ is $O(N^2)$. ■

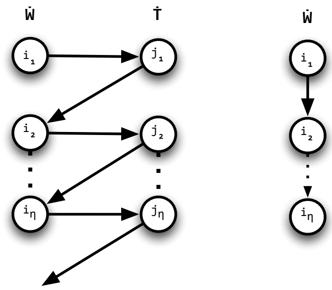
Lemma 4: Number of iterations in an acyclic admissible graph with η sources is bounded by $\eta(\eta + 1)/2$.

Proof: Let $\{1, 2, \dots, \chi_{\text{acyclic}}\}$ denote a sequence of successive NPR iterations and $\dot{G} = (\dot{W}, \dot{T}, \dot{A})$ the corresponding admissible graph. If this admissible graph consists of a single path $(i_1, j_1, i_2, j_2, \cdots, i_\eta, j_\eta)$, then the bound on iterations can be calculated using a directed tree as shown in Fig 6. If the admissible graph has multiple paths, then we have to form directed trees for each path and merge these trees to form a polytree as illustrated in Fig. 7. Let $\alpha(i)$ denote the number of ancestors of node i and $\boldsymbol{\alpha} \triangleq \{\alpha(i) | i = 1, 2, \dots, \eta\}$. Relabel the nodes such that $\boldsymbol{\alpha}$ is in descending order. We have this inequality

$$\alpha(i) \leq \eta - i$$

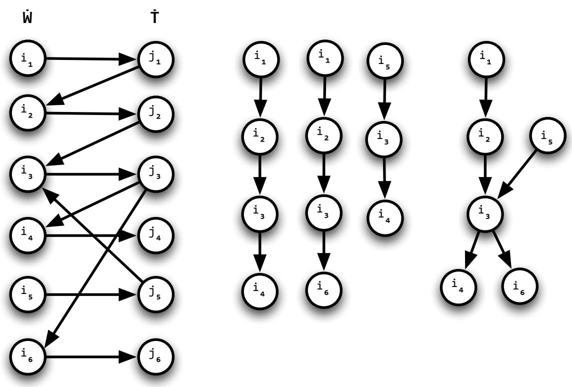
from the fact that in a directed acyclic graph if i is the ancestor of j, then j can't be the ancestor of i. χ_{acyclic} is bounded by

$$\begin{aligned} \sum_{i=1}^{\eta} \{\alpha(i) + 1\} &\leq \eta + \sum_{i=1}^{\eta} \{\eta - i\} \\ &= \eta(\eta + 1)/2. \end{aligned}$$



(a) Admissible Graph (b) Directed tree

Fig. 6: (a) Max. iterations on graph with one path. Assume that all sources have non-zero surpluses and we have the bidding sequence $i_\eta, i_{\eta-1}, i_\eta, i_{\eta-2}, i_{\eta-1}, i_\eta, \ldots, i_1, i_2, \ldots, i_\eta$. (b) shows the directed tree for this sequence. A node bids reversing the flow to the its child which allows the child to bid in the next iteration. So the max. number of times a node can bid is the number of its ancestors + 1. After a max. of $\eta(\eta+1)/2$ iterations all the sources in $\dot{W}$ are exhausted.



(a) (b) (c)

Fig. 7: (a) Admissible graph with 3 distinct paths. (b) directed trees corresponding to different paths. (c) Polytree obtained by merging. Max. number of times a node can bid is the number of its ancestors + 1.

■

Lemma 5: The number of iterations before the price drops by at least ϵ_{min} is $O(MN^3)$.

Proof: Let $\Delta = \{1, 2, \ldots, \chi\}$ denote this sequence of iterations. Every iteration is exhausting and the bids made during this sequence can't be reversed. So each sink j can only reverse flows of m_j bids which satisfy

$$p_j(1) \geq b_{i_1} \geq \cdots \geq b_{i_{m_j}} > p_j(1) - \epsilon_{min}$$

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

Let $n(t)$ be the number of sources with nonzero surpluses and its variation

$$\nabla n(t) = n(t+1) - n(t). \tag{4}$$

Let $\Delta^j \subset \Delta$ be the subsequence during which sink j is bid for. The subsequence of iterations during which $n(t)$ strictly increases is defined as

$$\Delta_+^j \triangleq \{t \in \Delta^j : \nabla n(t) > 0\}. \tag{5}$$

This happens if at least two flow reversals (one complete and one at least partial) take place during a given iteration. Since there are only m_j bids which can be reversed, we have

$$\#\Delta_+^j \leq \max\{m_j - 1, 0\}$$

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

$$\sum_{t \in \Delta_+^j} \{\nabla n(t)\} \leq m_j - 1. \tag{6}$$

Based on the what happens during the allocation phase at a sink node, an iteration can be classified as:

1) NPR: Flow is entirely accepted by reversal without any price drop. So

$$0 \leq \nabla n(t) \qquad \forall\ t \in \Delta_{\text{NPR}} \tag{7}$$

where $\Delta_{\text{NPR}} \subset \Delta$ is the subsequence of NPR iterations.

2) Price Reducing (PR): These are of two types:

a) Price Reducing with bid Reversal (PRR): These are strictly price reducing with some flow reversal.

$$0 \leq \nabla n(t) \qquad \forall\ n \in \Delta_{\text{PRR}}. \tag{8}$$

where $\Delta_{\text{PRR}} \subset \Delta$ is the subsequence of PRR iterations. This can only happen in two ways, a price drop followed by reversal (a bid starts getting reversed for the first time) or vice-versa (a bid is completely reversed). So each reversible bid can result at most 2 PRR iterations and

$$\#\Delta_{\text{PRR}}^j \leq 2m_j. \tag{9}$$

b) Price Reducing Without bid Reversal (PRWR): In such iterations, there is no reversal. So

$$\nabla n(t) = -1 \qquad \forall\ t \in \Delta_{\text{PRWR}} \tag{10}$$

where $\Delta_{\text{PRWR}} \subset \Delta$ is the subsequence of PRWR iterations.

From (5), (7), (8), and (10),

$$\Delta_{\text{NPR}}^j \bigcup \Delta_{\text{PRR}}^j = \Delta_+^j \bigcup \{t \in \Delta^j : \nabla n(t) = 0\}.$$

So using (6)

$$\sum_{t \in \Delta_{\text{NPR}}^j \bigcup \Delta_{\text{PRR}}^j} \nabla n(t) = \sum_{t \in \Delta_+^j} \nabla n(t) \leq m_j - 1. \tag{11}$$

Now we derive a bound for PRWRs using (4) and the fact that $n(\chi) > 0$,

$$\begin{aligned} -n(1) &\leq n(\chi) - n(1) \\ &= \sum_{t \in \Delta} \nabla n(t) \\ &= \sum_{t \in \Delta_{\text{PRR}} \bigcup \Delta_{\text{NPR}}} \nabla n(t) + \sum_{t \in \Delta_{\text{PRWR}}} \nabla n(t) \\ &\leq \sum_{j \in T} (m_j - 1) - \#\Delta_{\text{PRWR}}. \end{aligned}$$

TABLE I: Solution times (in seconds) for RAP Auction, Revised RAP Auction, CVX, and RWOA for solving randomly generated search problems. P_{edge} is the probability of an edge between a source and sink.

N	M	P_{edge}	Supply range	Gain range	ϵ	RAP	Revised RAP	CVX	RWOA
4	4	0.4	$1-7$	$0-2.3026$	0.01	0.00171	0.00219	0.15647	0.00130
4	4	0.4	$1-7$	$0-2.3026$	0.001	0.00602	0.00834	0.16417	0.00190
4	4	0.4	$1-7$	$0-2.3026$	0.0001	0.04508	0.06979	0.17767	0.00208
5	20	0.4	$1-7$	$0-2.3026$	0.001	0.04650	0.07731	0.98179	0.00372
20	5	0.4	$1-7$	$0-2.3026$	0.001	0.00404	0.00420	0.15637	0.00619
10	10	0.25	$1-10$	$0-2.3026$	0.001	0.01360	0.01949	0.17066	0.00480
10	10	0.5	$1-10$	$0-2.3026$	0.001	0.01587	0.02111	0.17749	0.00497
10	10	0.75	$1-10$	$0-2.3026$	0.001	0.02174	0.02819	0.15884	0.00370
10	10	1	$1-10$	$0-2.3026$	0.001	0.02960	0.03642	0.16027	0.00364
10	10	0.4	$1-10$	$0-2.3026$	0.0001	0.05600	0.07847	0.23108	0.01069
10	10	0.4	$1-10$	1	0.0001	1.26861	1.41890	-	0.00202
10	10	0.4	1	$0-2.3026$	0.001	0.01272	0.01752	0.34665	0.00870
10	10	0.4	$1-10$	$0-2.3026$	0.001	0.01345	0.01715	0.23284	0.00178
10	10	0.4	$1-100$	$0-2.3026$	0.001	0.00825	0.01030	0.23256	0.00510

Last inequality follows from (10) and (11). So

$$\#\Delta_{\text{PRWR}} \leq \sum_{j\in T}(m_j - 1) + n(1). \tag{12}$$

From (9) and (12) we have for all the PR iterations,

$$\begin{aligned}\#\Delta_{\text{PR}} &\triangleq \#(\Delta_{\text{PRR}} \bigcup \Delta_{\text{PRWR}})\\ &\leq \sum_{j\in T}\{3m_j - 1\} + n(1)\\ &\leq 3MN + N - 4M.\end{aligned}$$

From lemma 3, we have a bound on the number of successive NPRs iterations. So

$$\begin{aligned}\chi &\leq \#\Delta_{\text{PR}} * \text{max number of successive NPRs}\\ &\leq (3MN + N - 4M)O(N^2) = O(MN^3).\end{aligned}$$

■

Proposition 3: The revised RAP auction algorithm terminates in $O(N^3M^2p_{\max}c_{\max}/\epsilon)$ where $p_{\max} \triangleq \max_{j\in T} p_j(1)$ and $c_{\max} \triangleq \max_{(ij)\in E} c_{ij}$.

Proof: The sequence $p_j(t)$ is a non increasing sequence and lower and upper bounded by $p_{\min} \geq 0$ and $p_{\max}$, respectively. Every $O(MN^3)$ iterations the price of at least one sink node drops by $\epsilon_{\min}$. So maximum iterations is $O(N^3M^2p_{\max}/\epsilon_{\min})$. ■

The proofs for lemmas 3, 4 and 5 assume strict convexity. However this can be relaxed similar to how we generalized lemma 2 in [8].

V. EXPERIMENTS

In [1], we benchmarked the original RAP Auction for randomly generated instances of search theory problems [4] where the cost functions are exponentials. These tests were designed to study its performance relative to CVX [10], a MATLAB based generic convex solver, and Resource-Wise Optimization Algorithm (RWOA) [11] and the dependence of this performance on network topology, arc gains, supply and ϵ. In general, CVX was an order of magnitude slower than RAP Auction which was slower than RWOA. RWOA benefits from the fact that it has been customized for exponential cost functions.

Continuing along this line, we now report the performance for Revised RAP Auction. We have also improved the performance of original RAP Auction by efficient sorting. Table I lists the solutions times on MacBook Pro 4.1 running OS X 10.6.4 operating system. The performance of revised RAP Auction is only slightly more than the original RAP Auction.

VI. CONCLUSIONS

We have successfully proposed an extension to RAP Auction which enables it to have provably pseudo-polynomial complexity as opposed to finite termination. This extension takes the form of a single additional step to the bidding phase with negligible computation.

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] A. K. Bangla and D. A. Castañón, "RAP Auction : Auction algorithm for nonlinear resource allocation problems," 2010, CISE Report, Boston University.

[9] P. Tseng and D. P. Bertsekas, "An ϵ-relaxation method for separable convex cost generalized network flow problems," *Math. Prog.*, vol. 88, no. 1, pp. 85–104, June 2000.

[10] M. Grant and S. Boyd, "CVX: Matlab software for disciplined convex programming, version 1.21," http://cvxr.com/cvx, Aug. 2010.

[11] H. Luss and S. . K. Gupta, "Allocation of effort resources among competing activities," *Oper. Res.*, 1975.