Auction Mechanism for Multi-item Multi-attribute Outsourcing in Manufacturing Supply Networks

Abstract

We consider a multi-attribute outsourcing economy where both price and delivery date are relevant decision factors. In the economy, there is an Original Equipment Manufacturer (OEM) who needs multiple items to be delivered on the predetermined delivery dates and suppliers who are capable of delivering some or all of the items. We first show that there exists a competitive equilibrium in this economy. Then, an iterative auction mechanism is developed. The mechanism terminates at an equilibrium at which a near-efficient assignment is achieved. Finally, we prove the bound for the economy's inefficiency due to the mechanism.

Keywords: multi-item, multi-attribute, auction, outsource

1 Introduction

Outsourcing is a purchasing process designed to maximize delivered value. Companies may outsource products or services through a variety of trading mechanisms, such as negotiations, auctions, or any mix of these. Unfortunately, there is no trading mechanism which is suitable in all situations. Decisions have to be made as to which mechanism is to be used under which situation. Auctions work well in competitive environments, i.e., environments with many suppliers.

Bertsekas [1] introduce the distributed relaxation algorithm for the assignment problem, called the AUCTION algorithm. Even though the objective is to solve assignment problems efficiently in distributed manner, the algorithm operates like an iterative auction. Shapley and Shubik[7] analyzed the assignment game using primal-dual formulations analysis. They find that the outcomes in the core of the game are the solutions of the dual to the efficient assignment problem. Moreover, these dual solutions correspond to the equilibrium prices. Demange et al. [3] developed two iterative mechanisms for assignment problems, one achieves the exact minimum price equilibrium and another achieves the approximate one. Kelso and Crawford [4] studied a form of labor markets and generalized the result of Shapley and Shubik [7] by showing that Walrasian equilibrium in such model exists, provided that all workers are *gross substitutes* from each firm's standpoint. They also develop the salary-adjustment process that converges to the equilibrium biased in favor of agents on the side of the market that make offers. For exchange economy in which each buyer is allowed to buy a bundle of items, Parkes [6] developed a primal-dual-based iterative algorithm, called COMBAUCTION. The algorithm relies on the primal-dual formulations developed by Bikhchandani and Ostroy [2]. Note that all of the mechanisms mentioned above operate based on one attribute only (e.g., price).

This paper is organized as follows. Section 2 describes the model. Section 3 presents the linear programming formulations of the outsourcing economy. In section 4 and section 5, we provide the price interpretations of the variables in the linear programs and show that a competitive equilibrium exists in the economy. Section 6 proposes the auction mechanism and its inefficiency bound. Section 7 summarizes and concludes this paper.

2 The Model

We consider an outsourcing economy consisting of an OEM and a set of suppliers, $\mathcal{N} = \{1, \ldots, N\}$. The OEM has demand for a set of indivisible items $\mathcal{I} = \{1, \ldots, I\}$. Every item i in $\mathcal{I}$ has to be delivered on a date k in set $\mathcal{D} = \{1, \ldots, D\}$. A *bundle* $\mathcal{S}$ is a set of item(s) where $\mathcal{S} \subseteq \mathcal{I}$. We define a *tuple*, τ, by $\tau = (\mathcal{S}, (i, k_i) : \forall i \in \mathcal{S})$, where $\mathcal{S} \subseteq \mathcal{I}$ and k_i is the delivery date of item i. For example, let $\mathcal{N} = \{1, 2\}$, $\mathcal{I} = \{A, B\}$, $\mathcal{D} = \{Mon, Tue\}$. The possible tuples include $\tau_1 = (\{B\}, (B, Wed))$, $\tau_2 = (\{A, B\}, (A, Mon), (B, Tue))$, $\tau_3 = (\{A, B\}, (A, Tue), (B, Tue))$, and so on.

In cases where more number of attributes are required for decision-making, we can define tuple τ as: $\tau = (\mathcal{S}, (i, k_i, \ldots, l_i) : \forall i \in \mathcal{S})$, where $k_i, \ldots, l_i$ represent attributes' values of item i in $\mathcal{S}$. Our results can be easily applied to such cases as well.

The OEM has private valuation $v(\tau)$ on every tuple τ. The valuations are assumed non-negative. Similarly, for every tuple (τ), supplier j privately knows its production cost, which is denoted by $c_j(\tau)$. Again, the costs are assumed non-negative. In the case where a tuple is not desirable to the OEM (e.g. a bundle is definitely not desired on a particular delivery date for some reason), the OEM can set the tuple's valuation to zero. Similarly, a supplier may not be able to produce and deliver some tuple, in which case, the supplier can set the production cost of the tuple to infinity.

An *assignment*, represented by a vector $\alpha = \{(1, \tau_1), \ldots, (N, \tau_N)\}$ is an indication as to which supplier is assigned to which tuple. Let $\mathcal{S}_j$ be the bundle in tuple τ_j. An assignment α is *feasible* if a) all the bundles $\mathcal{S}_j$ for all $j \in \mathcal{N}$ are subsets of $\mathcal{I}$, b) all the delivery dates of the items in $\mathcal{S}_j$ for all $j \in \mathcal{N}$ are in $\mathcal{D}$, and c) none of the items is assigned to multiple suppliers: $\mathcal{S}_j \cap \mathcal{S}_l = \emptyset$, for all $j \neq l$. We denote a set of all feasible assignments by $\mathcal{A}$. A bundle that contains none of the items in $\mathcal{I}$ is called an *empty bundle*. Note that an assignment in which all of the suppliers are assigned to empty bundles is also considered a feasible assignment.

A tuple with empty bundle is called an *empty tuple*, and it is denoted by $\tau(\emptyset)$. An assignment in which all of the suppliers are assigned to empty tuples, is called an *empty assignment*, and it is denoted by $\alpha(\emptyset)$. Let $p(\tau)$ be the price of a tuple τ. The OEM wants to maximize its utility, or surplus, which is defined by: $U_c(\alpha) = \sum_{(j,\tau_j)\in\alpha} v(\tau_j) - p(\tau_j)$. Each supplier j in $\mathcal{N}$ wants to maximize utility (profit) which is defined by: $U_j(\tau) = p(\tau) - c_j(\tau)$.

We assume that OEM's valuation for the empty tuple is zero: $v(\tau(\emptyset)) = 0$ and supplier's cost of producing an empty tuple is zero: $c_j(\tau(\emptyset)) = 0$, for all $j \in \mathcal{N}$. An *economy surplus* is the sum of the utility of all the agents in the economy. We denote the economy described above by $\mathcal{E}(\mathcal{N}, \mathcal{I}, \mathcal{D}, U_c, U_j)$, or simply $\mathcal{E}$. Our objective is to find a stable outcome of $\mathcal{E}$ such that the economy surplus is maximized.

3 Linear Programming Formulations

A feasible assignment is said to be *efficient* if it maximizes the economy surplus, $U_{\mathcal{E}}$, over all possible assignments. An outcome of the economy describes who gets which item(s), if at all, and at what price. In this section we develop LP formulations that determine efficient assignments and find the prices that support the efficient assignments. We first introduce $\mathcal{LP}$, a primal problem, that determines efficient assignments.

$$(\mathcal{LP}) \quad V_{\mathcal{LP}} = \max \sum_{j\in\mathcal{N}} \sum_{\tau\in\mathcal{T}} [v(\tau) - c_j(\tau)] \cdot x_j(\tau) \tag{1}$$

subject to

$$\sum_{\alpha\in\mathcal{A}} y(\alpha) = 1 \tag{2}$$

$$x_j(\tau) \leq \sum_{\alpha \ni (j,\tau)} y(\alpha), \qquad \forall\, j \in \mathcal{N}, \forall \tau \in \mathcal{T} \tag{3}$$

$$\sum_{\tau\in\mathcal{T}} x_j(\tau) \leq 1, \qquad \forall\, j \in \mathcal{N} \tag{4}$$

$$x_j(\tau) \geq 0, \qquad \forall\, j \in \mathcal{N}, \forall \tau \in \mathcal{T} \tag{5}$$

$$y(\alpha) \geq 0, \qquad \forall\, \alpha \in \mathcal{A} \tag{6}$$

Variable $x_j(\tau)$ defines a tuple assignment (i.e., $x_j(\tau) = 1$ if a tuple τ is assigned to supplier j and $x_j(\tau) = 0$ otherwise). Variable $y(\alpha)$ indicates whether an assignment α is selected (i.e., $y(\alpha) = 1$, if an assignment α is selected and $y(\alpha) = 0$, otherwise).

The objective (1) is to maximize the economy surplus. Constraints (2) states that at most one tuple assignment can be selected. In constraint (3), we write $\alpha \ni (j, \tau)$ to mean "for all assignments that assign tuple τ to j." Through this set of constraints, we make sure that a tuple can be assigned to a particular supplier only if the selected assignment allows. Constraint (4) is in fact redundant. But, as will be clear in Section 5, they are necessary in developing the dual that gives useful information. Constraints (5) and (6)

restrict the variables to be non-negative. Let $(\mathbf{x}, \mathbf{y}) = ((x_j(\tau)), (y(\alpha))$ and $(\mathbf{x}^*, \mathbf{y}^*) = ((x_j^*(\tau)), y^*(\alpha))$ denote a feasible and an optimal solution, respectively.

Theorem 1 *$\mathcal{LP}$ has integral solution $(\mathbf{x}^*, \mathbf{y}^*)$.*

To derive the dual form of $\mathcal{LP}$, we need to identify the dual variables. Let π_c, $w_j(\tau)$, and π_j denote the dual variables associated with contraints (2), (3), and (4), respectively. The variable $w_j(\tau)$ represents the OEM's surplus due to the assignment of tuple τ to supplier j. Let $\mathbf{w}_j$ be the vector for $w_j(\tau)$ for all τ in $\mathcal{T}$. The two variables π_c and π_j respectively represent OEM's maximum surplus and supplier j's maximum profit, given $\mathbf{w}_j$ for all j in $\mathcal{N}$. Following is the dual of $\mathcal{LP}$, called $\mathcal{DLP}$.

$$(\mathcal{DLP}) \quad V_{\mathcal{DLP}} = \min \sum_{j \in \mathcal{N}} \pi_j + \pi_c \tag{7}$$

subject to

$$x_j(\tau): \qquad w_j(\tau) + \pi_j \geq v(\tau) - c_j(\tau), \qquad \forall\ j \in \mathcal{N}, \forall \tau \in \mathcal{T} \tag{8}$$

$$y(\alpha): \qquad \pi_c \geq \sum_{(j,\tau) \in \alpha} w_j(\tau), \qquad \forall\ \alpha \in \mathcal{A} \tag{9}$$

$$\pi_j \geq 0, \qquad \forall\ j \in \mathcal{N}, \forall \tau \in \mathcal{T} \tag{10}$$

Constraints (8) and (9) are associated with $\mathcal{LP}$'s variables $x_j(\tau)$ and $y(\alpha)$, respectively. A feasible solution of $\mathcal{DLP}$ is denoted by a vector $(\pi_c, (\pi_j), (\mathbf{w}_j))$.

A feasible solution to $\mathcal{LP}$ and a feasible solution to $\mathcal{DLP}$ are respectively optimal to $\mathcal{LP}$ and $\mathcal{DLP}$ if and only if the complementary slackness conditions are satisfied. For $\mathcal{LP}$-$\mathcal{DLP}$ pair, the complementary slackness conditions are:

$$[(w_j(\tau) + \pi_j) - (v(\tau) - c_j(\tau))] \cdot x_j(\tau) = 0, \quad \forall\ j \in \mathcal{N}, \forall \tau \in \mathcal{T} \tag{11}$$

$$[\pi_c - \sum_{(j,\tau) \in \alpha} w_j(\tau)] \cdot y(\alpha) = 0, \qquad \forall\ \alpha \in \mathcal{A} \tag{12}$$

$$[1 - \sum_{\tau \in \mathcal{T}} x_j(\tau)] \cdot \pi_j = 0, \qquad \forall\ j \in \mathcal{N} \tag{13}$$

These conditions are crucial for the proofs of our results in the subsequent sections.

4 Prices

It is interesting to note that all of the variables in $\mathcal{DLP}$ can be interpreted to represent profits, either to suppliers or OEM. None of them can be explicitly interpreted as the equilibrium price. However, price is typically the decision variable that gets adjusted in most trading mechanisms, including ours, in order for the mechanism to achieve the efficient assignment. In this section, we introduce price variables, which can also be used to describe the $\mathcal{DLP}$ feasible set.

There are two sets of constraints in $\mathcal{DLP}$, constraints (8) and (9). First consider constraints (8). Rearranging them, we have:

$$c_j(\tau) + \pi_j \geq v(\tau) - w_j(\tau), \qquad \forall\ j \in \mathcal{N}, \forall \tau \in \mathcal{T} \tag{14}$$

The left hand side of (14) can be interpreted as the price of a tuple τ at which supplier j achieves its maximum profit. We call it the *bid price*, $p_j^b(\tau)$, and it is defined by:

$$p_j^b(\tau) = c_j(\tau) + \pi_j \tag{15}$$

We interpret the right hand side of (14) to represent the price for a tuple τ that the customer is willing to pay to supplier j, given $w_j(\tau)$. We call it the *offer price*, $p_j^o(\tau)$, and it is defined by:

$$p_j^o(\tau) = v(\tau) - w_j(\tau) \tag{16}$$

Let $\mathbf{p}^b$ and $\mathbf{p}^o$, respectively, be the *bid price vector* and the *offer price vector* representing prices of all possible tuples for all suppliers. The prices interpreted from $\mathcal{DLP}$ are non-linear and non-anonymous. They are non-linear because prices of items are not additive. They are non-anonymous because prices for similar tuples, yet assigned to different suppliers, can be different.

OUTSOURCING

Set `stop = false`; $t = 0$; $\hat{\alpha} = \{\emptyset\}$; $\hat{\mathbf{p}}_j^o = \bar{\mathbf{p}}$, $\mathcal{B}_j^t = \emptyset, \forall\ j \in \mathcal{N}$;

`while(stop = false)` {

$t++$;

Each supplier j updates $\hat{\mathbf{p}}_j^b$ according to the price update rule shown in (17);

Each supplier j computes $\mathcal{B}_j$ through equation (18);

Consumer computes $\hat{\alpha}$, by solving TA, the formulation (19)–(22);

`if`$((\mathcal{B}_j^t = \emptyset,\ \forall\ j)$ or $(\mathcal{B}_j^t = \mathcal{B}_j^{t-1},\ \forall\ j)$ or $(\hat{\tau}_j \neq \tau(\emptyset),\ \forall\ j))$ `stop = true;`

`else` Consumer updates $\hat{\mathbf{p}}_j^o$, $\forall\ j \in \mathcal{N}$, through equation (23);

}

Figure 1: OUTSOURCING algorithm

5 Existence of Competitive Equilibrium

Competitive equilibrium (CE) is a desirable property for an economy. Before developing a mechanism for $\mathcal{E}$, it is essential to know wether a CE really exists. Relying on the connection between the complementary slackness conditions and the CE conditions (see, for example, Mas Colell et al. [5]), we derive the following result.

Theorem 2 *Let* $(\mathbf{p}^{o*}, \mathbf{p}^{b*})$ *be the price vector derived from* $(\pi_c^*, (\pi_j^*), (\mathbf{w}_j^*))$. *There exists a competitive equilibrium in* $\mathcal{E}$ *defined by* $(\mathbf{x}^*, \mathbf{y}^*)$ *and* $(\mathbf{p}^{o*}, \mathbf{p}^{b*})$.

6 Auction Mechanism

In this section, we develop a descending-price primal-dual-based auction mechanism, called OUTSOURCING, for the $\mathcal{LP}$-$\mathcal{DLP}$ pair. Then, we present the properties of the algorithm.

6.1 Proposed Algorithm

Given a primal feasible solution (i.e., a feasible assignment) and a dual feasible solution (i.e., a feasible profit vector), our algorithm iteratively updates prices by which the gap between feasible values $V_{\mathcal{LP}}$ and $V_{\mathcal{DLP}}$ is narrowed down. We assume that the price decrement, denoted by ϵ, is positive and constant. Let V_{OUT} be the value when OUTSOURCING terminates. We also present the bound for the discrepancy between V_{OUT} and $V_{\mathcal{LP}}^*$, where $V_{\mathcal{LP}}^*$ is the value of efficient assignment obtained from solving $\mathcal{LP}$.

In each iteration of OUTSOURCING, we need to know who is assigned to which tuple, and what are the offer prices and the bid prices at the moment. Let $\hat{\alpha} = \{(j, \hat{\tau}_j) \mid \forall\ j \in \mathcal{N}\}$ denote the *tentative assignment* in which tuple τ_j is assigned to supplier j. Also, let $\hat{\mathbf{p}}_j^o = (\hat{p}_j^o(\tau) \mid \forall \tau \in \mathcal{T})$ denote the *tentative offer price vector* prescribed by the OEM to supplier j and let $\hat{\mathbf{p}}_j^b = (\hat{p}_j^b(\tau) \mid \forall \tau \in \mathcal{T})$ denote the supplier j's *tentative bid price vector* in response to $\hat{\mathbf{p}}_j^o$. Of course, in a tentative assignment $\hat{\alpha}$, supplier j is assigned to a non-empty tuple $\hat{\tau}_j$ only if there exists a *tentative agreed price* $\hat{p}_j(\hat{\tau}_j)$ such that $\hat{p}_j(\hat{\tau}_j) = \hat{p}_j^o(\hat{\tau}_j) = \hat{p}_j^b(\hat{\tau}_j)$. Based on the $\mathcal{LP}$-$\mathcal{DLP}$ primal-dual pair, $\hat{\alpha}$ represents an $\mathcal{LP}$ solution while $\hat{\mathbf{p}}_j^o$ and $\hat{\mathbf{p}}_j^b$, for all $j \in \mathcal{N}$, represent a $\mathcal{DLP}$ solution.

Figure 1 illustrates the OUTSOURCING algorithm. At the beginning, all of the suppliers are assigned to empty tuples. The tentative offer price vectors for all suppliers are set equal to a starting price vector, $\bar{\mathbf{p}}$, containing $\bar{p}(\tau)$ for all τ. The iteration counter, t, is set to zero, and the bid sets of all the suppliers are set initially to empty sets.

After the initialization stage, we now enter the iterative process. As seen in the `while` loop of Figure 1, there are six main steps in each iteration. First, the iteration counter is incremented. Second, each supplier j updates the bid price vector $\hat{\mathbf{p}}_j^b$ through the following rule.

$$\hat{p}_j^b(\tau) = \begin{cases} \hat{p}_j^o(\tau), & \text{if } \hat{p}_j^o(\tau) \geq c_j(\tau) \\ \hat{p}_j^o(\tau) + \epsilon, & \text{if } \hat{p}_j^o(\tau) < c_j(\tau) \\ \infty, & \text{if } \bar{p}(\tau) < c_j(\tau) \end{cases} \tag{17}$$

This step ensures that each supplier will not incur a loss by accepting price that is lower than its cost. Additionally, if the starting price of a tuple is less than a supplier's cost, then the supplier will just ignore the tuple by setting a bid price at infinity. Third, with $\hat{\mathbf{p}}_j^b$, each supplier j calculates the bid set, $\mathcal{B}_j$, defined by

$$\mathcal{B}_j(\hat{\mathbf{p}}_j^b) = \{(\tau, \hat{p}_j^b(\tau)) \mid \hat{p}_j^b(\tau) - c_j(\tau) + \epsilon > \max\{0, \max_{\tau \in \mathcal{T}} \hat{p}_j^b(\tau) - c_j(\tau)\}\} \tag{18}$$

Note that $\mathcal{B}_j$ contains the tuple(s) that maximize supplier j's surplus (within ϵ accuracy). Each $\mathcal{B}_j$ is submitted to the OEM. Fourth, the OEM calculates the tentative assignment, $\hat{\alpha}$, that maximizes its surplus based on the bid sets submitted by the suppliers. This is accomplished by solving the Tentative Assignment model (TA) described by (19)–(22) as follows.

$$\max \sum_{j \in \mathcal{N}} [v(\tau) - \hat{p}_j^b(\tau)] \cdot z_j(\tau) \tag{19}$$

subject to

$$\sum_{\tau \in \mathcal{B}_j} z_j(\tau) \leq 1, \qquad \forall\ j \in \mathcal{N} \tag{20}$$

$$\sum_{j \in \mathcal{N}} \sum_{\mathcal{S} \ni i} z_j(\tau) \leq 1, \qquad \forall\ i \in \mathcal{I} \tag{21}$$

$$z_j(\tau) \in \{0, 1\}, \qquad \forall\ \tau \in \mathcal{B}_j, \forall\ j \in \mathcal{N} \tag{22}$$

The variable $z_j(\tau)$ indicates whether the tuple (τ), which is in $\mathcal{B}_j$, is assigned to supplier j. Fifth, the optimality conditions are verified. These conditions are i) none of the suppliers make profit for a given $\mathbf{p}^o$, ii) the bid sets of the all the suppliers do not change from the previous iteration, and iii) all of the suppliers are assigned to non-empty tuples. If any one of the three conditions is met, the algorithm stops. Sixth, if none of the conditions is met, the OEM then updates the tentative offer price vector, $\hat{\mathbf{p}}_j^o$, for all j in $\mathcal{N}$ that are assigned to empty tuples in that iteration, by setting:

$$\hat{p}_j^o(\tau) = \hat{p}_j^b(\tau) - \epsilon, \qquad \forall\ (\tau, \hat{p}_j^b(\tau)) \in \mathcal{B}_j, \forall j : \hat{\tau}_j = \tau(\emptyset) \tag{23}$$

Clearly, from iteration t to $t+1$, if a supplier repeats its bid price for a tuple, then OEM's offer price for that tuple also remains the same.

6.2 Properties of the Algorithm

Since prices are adjusted in a discrete step of ϵ, therefore V_{OUT} can be less than $V^*_{\mathcal{LP}}$. This implies that OUTSOURCING can cause inefficiency to the economy. We now present our first property of OUTSOURCING.

Proposition 1 OUTSOURCING *terminates in a finite number of iterations.*

Let $\hat{\pi}_c = \max_{\alpha \in \mathcal{A}} \sum_{(j,\tau) \in \alpha} v_j(\tau) - \hat{p}_j^b(\tau)$ and $\hat{\pi}_j = \max\{0, \max_{\tau \in \mathcal{T}} \hat{p}_j^o(\tau) - c_j(\tau)\}$ be OUTSOURCING's outputs in each iteration that correspond to π_c and π_j, respectively. The next three lemmas are used for proof of the inefficiency bound caused by OUTSOURCING.

Lemma 1 *In any iteration, if $\hat{\tau}_j$ is not equal to $\tau(\emptyset)$, then the following inequalities hold:*

$$\hat{p}^o(\hat{\tau}_j) - c_j(\hat{\tau}_j) + 2\epsilon \geq \hat{\pi}_j, \qquad \forall\ j \in \mathcal{N} \tag{24}$$

This lemma states that, in any iteration, if a supplier is assigned to a non-empty tuple, then that tuple maximizes his utility (within 2ϵ limit).

Lemma 2 *At termination, if $\hat{\tau}_j$ is equal to $\tau(\emptyset)$, then the following inequalities hold:*

$$\hat{p}^o(\tau) - c_j(\tau) \leq 0, \qquad \forall\ \tau \in \mathcal{T} \tag{25}$$

This lemma indicates that, at termination of OUTSOURCING, suppliers who receive no tuples are satisfied with the result because none of the tuples give positive profit.

Lemma 3 *In any iteration, the following inequality holds:*

$$\sum_{(j,\hat{\tau}_j) \in \hat{\alpha}} [v_j(\hat{\tau}_j) - \hat{p}_j^o(\hat{\tau}_j)] \geq \hat{\pi}_c \tag{26}$$

Inequality (26) indicates that, through OUTSOURCING, the OEM selects the bundle assignment that maximizes his surplus. This surplus is equal to its maximum surplus achievable.

From Lemma 1 to Lemma 3, we derive the following theorem.

Theorem 3 (Inefficiency Bound) *At termination, the value of the assignment obtained through the algorithm, V_{OUT}, satisfies the following:*

$$V_{\text{OUT}} + 2\min\{N, I\}\epsilon \geq V_{\mathcal{DLCP}} \geq V^*_{\mathcal{DLCP}} = V^*_{\mathcal{LCP}} \tag{27}$$

In words, the inefficiency caused by OUTSOURCING is less than or equal to $2\min\{N, I\}\epsilon$. Obviously, as ϵ approaches zero, the inefficiency approaches zero as well. Furthermore, if $\epsilon < \frac{1}{2\min\{N,I\}}$, then the algorithm always achieves efficient assignment.

7 Conclusion

In this paper, we consider a multi-attribute outsourcing economy where both price and delivery date are relevant decision factors. The economy consists of one OEM and multiple suppliers. We show that there exists a competitive equilibrium in this economy. Then, we develop a primal-dual-based auction mechanism for the economy. We finally prove the inefficiency bound for the mechanism.

References

[1] D. P. Bertsekas. A distributed algorithm for the assignment problem. Working paper, Laboratory for Information and Decision Systems, M.I.T., March 1979.

[2] Sushil Bikhchandani and Joseph M. Ostroy. The package assignment model. Technical report, Anderson School of Management, University of California, Los Angeles, June 2001.

[3] Gabrielle Demange, David Gale, and Marilda Sotomayor. Multi-item auctions. *Journal of Political Economy*, 94(4):863–872, 1986.

[4] Alexander S. Kelso and Vincent P. Crawford. Job matching, coalition formation, and gross substitutes. *Econometrica*, 50(6):1483–1504, 1982.

[5] Andreu Mas-Colell, Michael D. Whinston, and Jerry R. Green. *Microeconomic Theory*. Oxford University Press, Oxford, 1995.

[6] David C. Parkes. Iterative combinatorial auctions: Achieving economic and computational efficiency. Dissertation, University of Pennsylvania, 2001.

[7] Lloyd S. Shapley and Martin Shubik. The assignment game i: The core. *International Journal of Game Theory*, 1(2):111–130, 1972.