

Chapter 4

Linear Programming and Auction Design

Mechanism design proposes the Vickrey-Clarke-Groves mechanism as a solution for the combinatorial allocation problem. In fact, this mechanism is essentially the *only* mechanism with the critical properties of strategy-proofness and allocative-efficiency. However, we have identified a number of inherent computational problems with Groves mechanisms:

- Every agent must provide its value for every bundle to the auctioneer.
- The auctioneer must solve multiple NP-hard problems to compute the outcome of the auction.

The Groves mechanisms are *centralized* solutions to a decentralized optimization problem. They address the incentive issues in systems with distributed agents with private information but fail to address important computational issues.

Naive approaches to address the auctioneer's computational problem will often break the strategy-proofness of the mechanism, in addition to reducing the allocative-efficiency of the solution. One method discussed in Chapter 3 suggests computing approximate solutions to the winner-determination problems, and computing Vickrey payments with the approximate solutions. Strategy-proofness is lost as soon as the mechanism does not compute the efficient allocation, the allocation that maximizes the reported values of agents. Instead an agent should try to misrepresent its valuation function *in just the right way* to make the auctioneer compute the optimal winner-determination solution *despite* its approximate algorithm. In other words agents should try to compensate for the approximation within the mechanism.

Similarly, while it is possible to restrict a bidding language such that the mechanism's winner-determination problem is tractable (see Table 4.4), as soon as the restricted expressiveness of the language forces the agent to submit an approximate report of its valuation

function, the agent must reason about *which approximation* will lead to the auctioneer computing a solution to the winner-determination problem that is maximal for the agent's true valuation function.

The motivation for *iterative* combinatorial auctions is to address the first computational problem, that of agent valuation work, *and* retain the useful game-theoretic properties of *strategy-proofness* and *efficiency*. In many problems it is quite unrealistic to assume that an agent can compute its value for all possible combinations of items, as is required in the single-shot VCG mechanism. Iterative combinatorial auctions provide *interactive* solutions, hopefully requesting *just enough* information from agents to compute the efficient allocation and the Vickrey payments. The uniqueness of Groves mechanisms (amongst direct-revelation mechanisms) implies via the revelation principle that any iterative solution to the combinatorial allocation problem with these desirable game-theoretic properties *must* compute the payments in the Vickrey-Clarke-Groves mechanism.

My approach is to first *assume* a simple bidding strategy for agents in each round of an iterative auction. The strategy, *myopic best-response*, need not be game-theoretically rational for an agent. However, this assumption allows a strong connection between linear programming theory, in particular primal-dual algorithms, and iterative auction design. Chapters 4 and 5 introduce primal-dual algorithm COMBAUCTION, and its auction equivalent *iBundle*, which computes efficient allocations with myopic best-response agent strategies. The prices computed in the dual solution have an economic interpretation, as the *competitive equilibrium* prices. Later, in Chapters 6 and 7, I present an extended primal-dual method, VICKAUCTION, and an experimental auction design, *iBundle Extend&Adjust*, to compute Vickrey payments and the efficient allocation with myopic best-response agent strategies. Vickrey payments make myopic best-response a *sequentially rational* strategy for an agent in equilibrium with myopic best-response from other agents—justifying my earlier assumption. This “LP + myopic best-Response + Vickrey” approach appears to provide a compelling methodology for the design of iterative mechanisms with useful game-theoretic properties.

Bertsekas [Ber87] had earlier proposed a primal-dual algorithm AUCTION for the *assignment problem*, which is a special case of the combinatorial allocation problem. AUCTION has a natural interpretation as an ascending-price auction, but does not compute Vickrey payments and does not have any useful incentive-compatibility properties. As discussed

in Section 4.7, authors such as Demange *et al.* [DGS86] and Ausubel [Aus97, Aus00], have proposed iterative auctions that compute both the efficient allocation and Vickrey payments for special-cases of the combinatorial allocation problem (CAP). In *iBundle* I extend Bertsekas' primal-dual methodology to solve the CAP, without placing any restrictions on agent preferences, but for the moment without computing Vickrey payments (see Chapters 6 and 7 for this extension). *iBundle* implements a primal-dual algorithm COMBAUCTION, which computes solutions to linear program formulations of the CAP introduced in Bikchandani & Ostroy [BO99].

The extended auction, *iBundle Extend&Adjust*, implements a primal-dual algorithm, VICKAUCTION for a new linear program formulation of the Vickrey payments. VICKAUCTION provably computes the efficient allocation and Vickrey payments with best-response information from agents. Computational results in Chapter 7 demonstrate that the experimental auction, *iBundle Extend&Adjust*, which is an interpretation of primal-dual method VICKAUCTION, computes Vickrey payments over a suite of problem instances. A full proof of the extended auction awaits a proof of termination of its final phase (see Chapter 7).

The outline of this chapter is as follows. Section 4.1 presents a brief description of *iBundle*. Section 4.2 provides background on linear programming theory and primal-dual algorithms. Section 4.3 relates primal-dual methods with allocation problems, and considers price-adjustment methods and competitive equilibrium. The English auction provides a simple primal-dual example. Section 4.4 provides a hierarchy of linear programming formulations for the combinatorial allocation problem. Section 4.5 also outlines the tractable special-cases of the combinatorial allocation problem, and to provide practical interpretations as much as possible. This falls naturally within this chapter because the tractable special cases can all be understood within linear programming theory.

Section 4.6 describes COMBAUCTION, a primal-dual algorithm for the combinatorial allocation problem. *iBundle*, introduced in the next chapter, is a straightforward auction interpretation of COMBAUCTION. Finally, Section 4.7 compares the characteristics and properties of COMBAUCTION and *iBundle* with earlier iterative auction mechanisms.

4.1 Overview: The *i*Bundle Auction

*i*Bundle [Par99, PU00a] is an ascending-price combinatorial auction, in which prices are maintained on bundles of items and agents can bid for bundles of items directly. In this section I give only a high-level description of the auction. Full details are presented in the next chapter. The description here is included to give some context to the primal-dual method, COMBAUCTION, introduced to solve the CAP.

Allowing bids on bundles of items allows an agent to express a “I only want A if I also get B” type of constraint, that is observed to be important in many applications (see Chapter 1).

*i*Bundle maintains an ask price, $p_i(S) \geq 0$ for every bundle $S \subseteq \mathcal{G}$ and every agent $i \in \mathcal{I}$. This is the minimal price that an agent must bid for that bundle in the current round of the auction. The prices may be non-linear, $p_i(S) \neq \sum_{j \in S} p_i(j)$, and may be non-anonymous, $p_i(S) \neq p_j(S)$. In practice it is not necessary to explicitly price every bundle, prices are explicitly maintained on a subset of bundles (those which receive unsuccessful bids) and can be computed on any bundle as necessary. The auction also maintains a provisional allocation, $S = (S_1, \dots, S_I)$ in each round. This is adjusted across rounds in response to agents’ bids until the auction terminates, when it is implemented as the final allocation.

The key components of *i*Bundle are the bidding language, the winner-determination rules, the price-update rules, and the termination conditions. Agents place exclusive-or bids for bundles, e.g. $S_1 \text{ XOR } S_2$, to indicate that an agent wants either all items in S_1 or all items in S_2 but not both S_1 and S_2 . Each bid is associated with a bid price, which must be at least the ask price for the bundle. The auctioneer collects bids and computes a provisional allocation to maximize revenue given the bids. If every agent that placed a bid receives a bundle the auction terminates. Prices are initially anonymous, with $p_{\text{ask},i}(S) = p_{\text{ask},j}(S) = p(S)$, but a simple rule introduces price discrimination dynamically, as necessary to terminate with competitive equilibrium prices and an efficient allocation. The price on a bundle is increased to $\epsilon > 0$ above the highest bid price for the bundle from any unsuccessful agent in the current round (an agent not in the provisional allocation). *i*Bundle terminates when every agent that bids at the current prices receives a bundle in the provisional allocation.

The final prices are competitive equilibrium prices, and the final allocation efficient, if

agents follow a myopic best-response bidding strategy, bidding for bundles that maximize their utility given the prices in each round.

4.2 Linear Programming Theory

First, I provide a brief review of basic results in linear programming. See Papadimitriou & Steiglitz [PS82] for a text book introduction, and Chandru's excellent survey papers [CR99b, CR99a] for a modern review of the literature on linear programming and integer programming.

Consider the linear program:

$$\begin{aligned} \max \quad & c^T x && \text{[P]} \\ \text{s.t.} \quad & Ax \leq b \\ & x \geq 0 \end{aligned}$$

where A is a $m \times n$ integer matrix, $x \in R^n$ is a n -vector, and c and b are n - and m -vectors of integers. vectors are column-vectors, and notation c^T indicates the *transpose* of vector c , similarly for matrices. The primal problem is to compute a feasible solution for x that maximizes the value of the objective function.

The dual program is constructed as:

$$\begin{aligned} \min \quad & b^T y && \text{[D]} \\ \text{s.t.} \quad & A^T y \geq c \\ & y \geq 0 \end{aligned}$$

where $y \in R^m$ is a m -vector. The dual problem is to compute a feasible solution for y that minimizes the value of the objective function.

Let $V_{LP}(x) = c^T x$, the value of feasible primal solution x , and $V_{DLP}(y) = b^T y$, the value of feasible dual solution y .

The *weak duality theorem* of linear programming states that the value of the dual always dominates the value of the primal:

THEOREM 4.1 (weak-duality). *Given a feasible primal solution x with value $V_{LP}(x)$ and a feasible dual solution y with value $V_{DLP}(y)$, then $V_{LP}(x) \leq V_{DLP}(y)$.*

PROOF. Solution x is feasible, so $Ax \leq b$. Solution y is feasible, so $A^T y \geq c$. Therefore, $x \leq A^T b$ and $y \geq Ac$, and $c^T x \leq c^T A^T b = b^T AC \leq b^T y$, and $P \leq D$. ■

The *strong duality theorem* of linear programming states that primal and dual solutions are optimal if and only if the value of the primal equals the value of the dual:

THEOREM 4.2 (strong-duality). *Primal solution x^* and dual solution y^* are a pair of optimal solutions for the primal and dual respectively, if and only if x^* and y^* are feasible (satisfy respective constraints) and $V_{LP}(x^*) = V_{DLP}(y^*)$.*

The strong-duality theorem of linear programming can be restated in terms of *complementary-slackness* conditions (CS for short). Complementary-slackness conditions expresses logical relationships between the values of primal and dual solutions that are necessary and sufficient for optimality.

DEFINITION 4.1 [complementary-slackness] Complementary-slackness conditions constrain pairs of primal and dual solutions. *Primal* CS conditions state $x^T(A^T y - c) = 0$, or in logical form:

$$x_j > 0 \Rightarrow A^j y = c_j \quad (\text{P-CS})$$

where A^j denotes the j th column of A (written as a row vector to avoid the use of transpose). *Dual* CS conditions state $y^T(Ax - b) = 0$, or in logical form:

$$y_j > 0 \Rightarrow A_i x = b_i \quad (\text{D-CS})$$

where A_i denotes the i th row of A .

The strong-duality theorem can be restated as the *complementary-slackness theorem*:

THEOREM 4.3 (complementary-slackness). *A pair of feasible primal, x , and dual solutions, y , are primal and dual optimal if and only if they satisfy the complementary-slackness conditions.*

PROOF. P-CS iff $x^T(A^T y - c) = 0$, and D-CS iff $y^T(Ax - b) = 0$. Equating, and observing that $x^T A^T y = y^T Ax$, we have P-CS and D-CS iff $x^T c = y^T b$, or $c^T x = b^T y$. The LHS is the value of the primal, $V_{LP}(x)$, and the RHS is the value of the dual, $V_{DLP}(y)$. By the strong duality theorem, $V_{LP}(x) = V_{DLP}(y)$ is a necessary and sufficient condition for the solutions to be optimal. ■

4.2.1 Primal-Dual Algorithms

Primal-dual is an algorithm-design paradigm that is often used to solve combinatorial optimization problems. A problem is first formulated both as a primal and a dual linear program. A primal-dual algorithm searches for feasible primal and dual solutions that satisfy complementary-slackness conditions, instead of searching for an optimal primal (or dual) solution directly. Primal-dual can present a useful algorithm-design paradigm for combinatorial optimization problems. Instead of solving a single hard primal solution, or a single hard dual solution, a primal-dual approach solves a sequence of restricted primal problems. Each restricted primal problem is often much simpler to solve than the full primal (or dual) problem [PS82].

Primal-dual theory also provides a useful conceptual framework for the design of iterative combinatorial auctions. Prices represent a feasible dual solution, and bids from agents allow a search for a primal solution that satisfies complementary-slackness conditions. If the current solution is suboptimal there is enough information available to adjust dual prices in the right direction. Complementary-slackness conditions provide the key to understanding how it is possible to compute and verify optimal solutions without complete information: it is sufficient to just verify that a feasible solution satisfies CS conditions. Primal-dual algorithms are consistent with the decentralized information inherent in distributed agent-based systems. Optimality reduces to a test of feasibility and complementary-slackness, which is available from agent bids, rather than the direct solution of a primal problem, which requires information about agent valuation functions.

A standard primal-dual formulation maintains a feasible dual solution, y , and computes a solution to a *restricted primal problem*, given the dual solution. The restricted primal is formulated to compute a primal solution that is both feasible and satisfies CS conditions with the dual solution. In general this is not possible (until the dual solution is optimal),

and a relaxed solution is computed. The restricted primal problem is typically formulated to compute this relaxed solution in one of two ways:

1. Compute a feasible primal solution x' that minimizes the “violation” of complementary-slackness conditions with dual solution y .
2. Compute a primal solution x' that satisfies complementary slackness conditions with dual solution y , and minimizes the “violation” of feasibility constraints.

Method (1) is more useful in the context of iterative auction design because it maintains a feasible primal solution, which becomes the *provisional allocation* in the auction, i.e. a tentative allocation that will be implemented only when the auction terminates. The restricted primal problem can be solved as a *winner-determination problem*. I show that computing the allocation that maximizes revenue given agent bids (the solution to winner-determination) is a suitable method to minimize the violation of CS conditions between the prices and the provisional allocation in each round *i*Bundle. Prices in each round of an auction define the feasible dual solution, and agent best-response bids provide enough information to test for complementary-slackness and adjust solutions towards optimality.

As discussed in the introduction to this chapter, I first assume myopic best-response, but later justify this assumption with an extension to compute Vickrey payments at the end of the auction in addition to the efficient allocation (see Chapters 6 and 7).

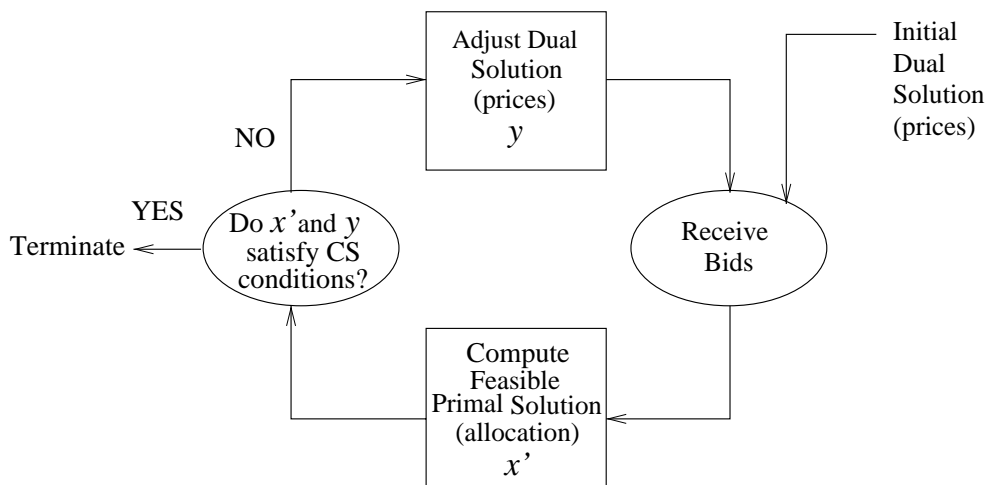


Figure 4.1: A primal-dual interpretation of an auction algorithm.

A primal-dual based auction method has the following form (see Figure 4.1):

1. Maintain a feasible dual solution (“prices”).
2. Compute a feasible primal solution (“provisional allocation”) to minimize violations with complementary-slackness conditions given agents’ bids.
3. Terminate if all CS conditions are satisfied (“are the allocation and prices in competitive equilibrium?”)
4. Adjust the dual solution towards an optimal solution, based on CS conditions and the current primal solution (“increase prices based on agent bids”)

4.3 Allocation Problems

Let us consider the particular form of an *allocation problem*, in which there are a set of discrete items to allocate to agents, and the goal is to maximize value. We assume quasi-linear preferences, and use *utility* to refer to the difference between an agent’s value for a bundle and the price. The primal and dual allocation problems can be stated as follows:

DEFINITION 4.2 [allocation problem: primal] The primal allocation problem is to allocate items to agents to maximize the sum value over all agents, such that no item is allocated to more than one agent.

DEFINITION 4.3 [allocation problem: dual] The dual allocation problem is to assign *prices* to items, or bundles of items, to minimize the sum of (i) each agents’ maximum utility given the prices, over all possible allocations; *and* (ii) the maximum revenue over all possible allocations given the prices.

Clearly, without information on agents’ values the auctioneer cannot compute an optimal primal or an optimal dual (because of term *(i)* in the dual). However, under a reasonable assumption about agents’ bidding strategies (myopic best-response) the auctioneer can verify complementary-slackness conditions between primal and dual solutions, and adjust prices and the allocation towards optimal solutions.

An auction interpretation of the complementary-slackness conditions can be stated as follows:

DEFINITION 4.4 [allocation problem: CS conditions] The CS between a feasible primal solution to an allocation problem, x , and a feasible dual solution, prices p , are:

- (CS-1) Agent i receives bundles S_i in the provisional allocation if and only if the bundle maximizes its utility given the prices, and has non-negative utility.
- (CS-2) The provisional allocation $S = (S_1, \dots, S_I)$ is the revenue-maximizing allocation given the prices.

Left deliberately vague at this stage is the exact *structure* of the prices. In a combinatorial allocation problem these might need to be non-linear and non-anonymous prices to support the optimal allocation. Similarly, the revenue-maximization concept must be defined with respect to a particular linear program formulation. Note also that CS-2 is not automatically satisfied with a provisional allocation computed to maximize revenue given agents' bids. CS-2 makes a stronger claim, that the provisional allocation must maximize revenue over *all possible* allocations given the current ask prices, not just over all allocations consistent with bids.

Primal-dual auction analysis requires the following assumption about agent strategies:

DEFINITION 4.5 [myopic best-response] A myopic best-response bidding strategy is to bid for all items or bundles of items that maximize utility at the current prices.

Best-response bids provide enough information to test CS-1, because the best-response of an agent is precisely those bundles that maximize an agent's utility given the current prices. For any feasible primal solution, the auctioneer can test CS-2 because that only requires price information.

The restricted primal has a natural auction interpretation:

DEFINITION 4.6 [auction restricted-primal problem] Given best response bids from each agent allocate bundles to maximize revenue, breaking ties in favor of including more agents in the provisional allocation.

Note well that a bundle is only allocated to an agent in the restricted primal problem if the agent bids for that bundle. This restriction ensures that CS-1 is satisfied for that agent, given the definition of myopic best-response. CS-2 is satisfied with careful price-adjustment rules, such that prices are increased "slowly enough" that the revenue-maximizing allocation can always be computed from agent bids.

Given myopic best-response, the termination condition, which tests for complementary-slackness between the provisional allocation and the prices, must check that CS-1 holds

for every agent. This is achieved when every agent to submit a bid receives a bundle in the provisional allocation, i.e. in competitive equilibrium.

Our interest is in solving the CAP, which is most immediately formulated as an integer program (see Section 4.4). In order to apply primal-dual methods it is essential that we have a linear program formulation of the CAP. We must be careful enough to use a strong enough formulation, such that the optimal solution is integral (0-1) and not fractional. The ideal situation is illustrated in Figure 4.2. The auction implements a primal-dual algorithm for a linear program that is strong enough to compute the optimal integer solution.

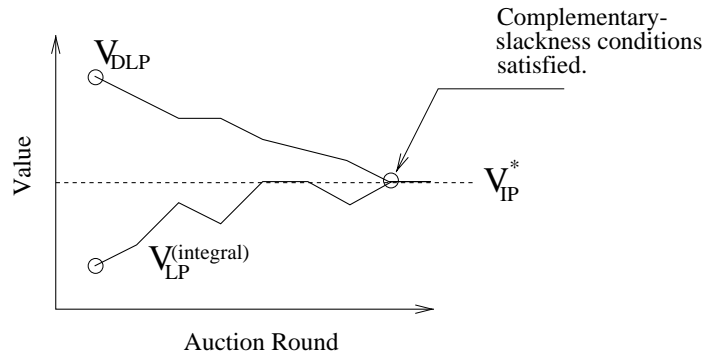


Figure 4.2: An auction-based primal-dual algorithm in which the linear program formulation is strong enough to eliminate all fractional solutions.

In comparison, consider Figures 4.3 (a) and (b), which illustrate a primal-dual algorithm and iterative auction method for a linear program that is not strong enough, and admits optimal fractional solutions. The primal-dual algorithm terminates with a fractional primal solution and value greater than the value of the best possible integer solution. The auction always maintains an integral primal solution (solving winner-determination to compute the provisional allocation), but can terminate with a primal solution that does not satisfy complementary-slackness conditions. Although the primal solution is perhaps optimal, its optimality cannot be assessed without CS information.

4.3.1 Price Adjustment

Left undefined at the moment, and the challenging part of primal-dual auction design, are the precise rules used to define price updates. The goal is to use information from agents' bids, and the current provisional allocation, to adjust prices towards an optimal dual solution—that will support an optimal primal solution. Primal-dual methods traditionally

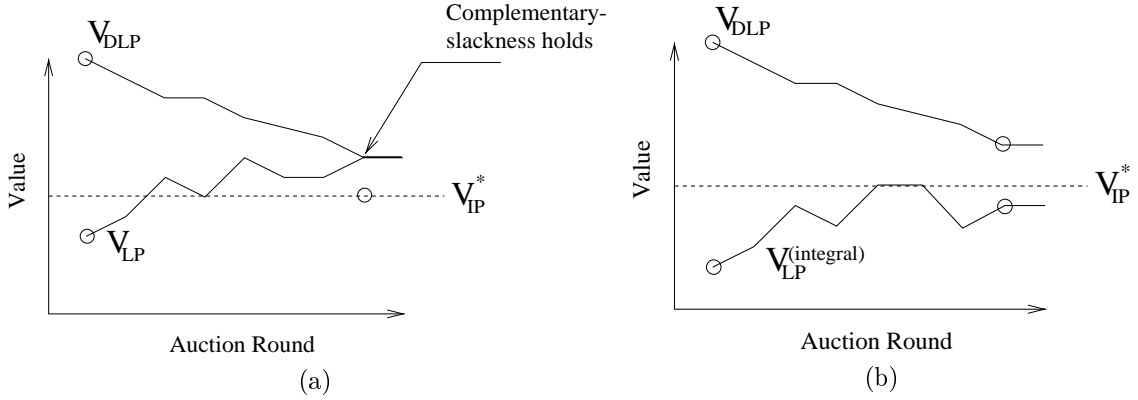


Figure 4.3: Primal-dual algorithm (a) and Primal-dual auction method (b) in which the linear program relaxation is too weak, and $V_{LPR}^* > V_{IP}^*$.

use the dual of the restricted primal to adjust the dual solution across iterations. A simpler method in allocation problems is to *increase prices on over-demanded items, or bundles of items*. The method can be explained both in terms of its effect on complementary-slackness conditions and in terms of its effect on the value of the dual solution.

The idea is to increase prices to: (a) maintain CS-2 in the next round and (b) move towards satisfying CS-1 for all agents.

PROPOSITION 4.1 (progress). *Progress is made towards satisfying CS-1 and CS-2 with the provisional allocation and the ask prices if: (1) the auctioneer increases prices on one or more bundles that receive bids in each round; and (2) the auctioneer increases prices by a small enough increment that best-response bids from agents continue to maximize revenue in the next round.*

CS-1 holds whenever every agent that bids receives a bundle in the provisional allocation. This is trivially achieved for high enough prices because no agent will bid, but we need to achieve this condition in combination with CS-2. The trick is to increase prices just enough to maintain revenue-maximization from bids CS-2 across all rounds. This is achieved in *iBundle* by ensuring that myopic agents continue to bid for bundles at the new prices, i.e. increasing price on over-demanded bundles.

An alternative interpretation is that increasing prices on over-demanded items will *reduce the value of the dual*, making progress towards the optimal solution, see Figure 4.4. Recall that the value of the dual is the sum of the auctioneer's maximal revenue and each

agent’s maximal utility at the current prices. A price increase will decrease the value of the dual if the increase in maximal revenue from the price increase is *less* than the decrease in total maximal utility summed across agents.

The auctioneer can achieve this effect of increasing revenue by less than the decrease in agent utility by selecting over-demanded items, or bundles of items, on which to increase the price. Suppose that two agents bid for bundle S_1 , and that both agents have at least $\epsilon > 0$ more utility for that bundle than any other bundle at the current prices. Increasing the price on over-demanded bundle S_1 by ϵ will decrease the maximal utility of *both* agents by ϵ , for a decrease in dual value of 2ϵ . However, increasing the price on this one bundle by ϵ can increase the auctioneer’s maximal revenue by at most ϵ . The result is that the net change in utility must a decrease of at least ϵ .

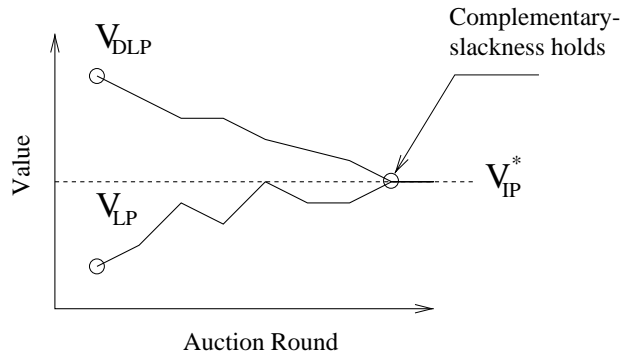


Figure 4.4: Primal-dual interpretation of an ascending-price auction.

4.3.2 Competitive Equilibrium

The optimal primal and dual solutions in an allocation problem correspond to a classic statement of *competitive equilibrium*.

DEFINITION 4.7 [competitive equilibrium] Allocation S and prices p are in competitive equilibrium when:

- (a) every agent receives a bundle in its best-response (utility maximizing) set
- (b) the allocation maximizes the revenue for the auctioneer at the prices

The allocation in competitive equilibrium is efficient, by equivalence between competitive equilibrium and primal-dual optimality:

THEOREM 4.4 (competitive equilibrium efficiency). *An allocation S is efficient if and*

only if there exists competitive equilibrium prices p , for an appropriate type of prices (e.g. linear, bundle, non-anonymous).

In the context of the combinatorial allocation problem Bikchandani & Ostroy [BO99] have characterized the structure on prices required for the existence of competitive equilibrium (and equivalently for integral solutions to linear program formulations of CAP). These formulations are introduced in Section 4.4 and discussed at length.

In some problems it is necessary that prices are both non-linear (bundle prices) and non-anonymous (different prices for the same bundle to different agents) to support a competitive equilibrium solution.

Wurman & Wellman [WW99, WW00] propose an alternative definition of competitive equilibrium, which is essentially complementary slackness condition CS-1 without CS-2. This relaxed condition is sufficient for the existence of equilibrium prices even without non-anonymous prices, but too weak to be able to claim that equilibrium prices imply an efficient allocation.

4.3.3 Example: The English Auction

The standard English auction illustrates the primal-dual framework for auction design. The English auction is an ascending-price auction for single items, where the price increases as long as more than one agent bids at the current price.

Let v_i denote agent i 's value for the item. The single-unit resource allocation problem is:

$$\begin{aligned} & \max \sum_i v_i x_i && \text{[IP}_{\text{single}}\text{]} \\ \text{s.t.} & \sum_i x_i \leq 1 \\ & x_i \in \{0, 1\} \end{aligned}$$

where $x_i = 1$ if and only if agent i is allocated the item, i.e. the goal is to allocate the item to the agent with the highest value. This can be solved as a linear program, [LP_{single}], relaxing the integral constraint

$$\begin{aligned}
& \max \sum_i v_i x_i && [\text{LP}_{\text{single}}] \\
\text{s.t. } & \sum_i x_i \leq 1 \\
& x_i \geq 0
\end{aligned}$$

and $V_{\text{LP}}^* = V_{\text{IP}}^*$, i.e. there is always an integral optimal solution to the relaxed problem. The dual formulation, $[\text{DLP}_{\text{single}}]$, is

$$\begin{aligned}
& \min \pi && [\text{DLP}_{\text{single}}] \\
\text{s.t. } & \pi \geq v_i, \quad \forall i \\
& \pi \geq 0
\end{aligned}$$

The complementary-slackness conditions are

$$\begin{aligned}
\sum x_i \geq 0 & \Rightarrow \pi = v_i, \quad \forall i \\
\pi > 0 & \Rightarrow \sum x_i = 1
\end{aligned}$$

The complementary-slackness conditions can be interpreted in terms of competitive equilibrium conditions on the allocation and the prices. An allocation and prices in a single-item auction are in competitive equilibrium, and the allocation is efficient, when:

(i) the item is sold to an agent, that agent bids for the item at the price, and no other agent bids for the item at the price.

or (ii) the item is sold to no agent, the price is zero, and no agent bids for the item.

It is straightforward to understand efficiency in these cases: in (i) the agent with the highest value receives the item; in (ii) no agent has a positive value for the item.

The English auction maintains price p on the item, initially $p = 0$. Agent i bids whenever $p < v_i$, and the provisional allocation sets $x_j = 1$ for one of the agents that bids in each round, and increases the price p whenever more than one agent bids.

Let the provisional allocation define a feasible primal solution, and the price define dual solution $\pi = \sum_i \max\{0, v_i - p\} + p$. This is feasible, $\pi \geq \max\{0, v_i - p\} + p \geq v_i$ for all agents i .

Assume that agents follow a myopic best-response bidding strategy, bidding for the item at the ask price whenever the price is below their value. The optimality of the English auction can be understood in two different ways:

- The English auction terminates with primal and dual solutions that satisfy CS-1 and CS-2.

Clearly, CS-2 is satisfied throughout the auction because the item is always allocated to one of the agents. CS-1 is satisfied when the auction terminates. Let j indicate the only agent that bids at price p . Therefore $v_i - p \leq 0$ for all agents $i \neq j$ and $v_j - p \geq 0$ for agent j (because agents follow best-response bidding strategies), and $\pi = \sum_i \max\{0, v_i - p\} + p = \max\{0, v_j - p\} + p = v_j$.

- The value of the dual strictly decreases in each round of the auction. Let $m > 1$ equal the number of agents that bid in each round of the auction except the final round. For price increment ϵ , the sum maximal utility to the agents decreases by $m\epsilon$ and the maximal revenue to the auctioneer increases by ϵ , for a net change in π of $-(m - 1)\epsilon$.

In fact, the final price in the English auction approaches the Vickrey payment (i.e. the second-highest value) as the bid increment $\epsilon \rightarrow 0$. It follows that myopic-best response is a rational *sequential* strategy for an agent, in equilibrium with myopic best-response strategies from other agents (see Chapter 7 for a full discussion of the incentive properties of iterative Vickrey auctions).

4.4 Linear Program Formulations for the Combinatorial Allocation Problem

Primal-dual based auction methods require linear programming formulations of allocation problems. Bikchandani & Ostroy [BO99] have formulated a hierarchy of linear programs for the problem, introducing additional constraints to remove fractional solutions. Although it is always possible to add enough constraints to a linear program relaxation to make the optimal solution integral [Wol81a, Wol81b, TW81], the particular formulations proposed by Bikchandani & Ostroy are interesting because the constraints have natural interpretations as prices in the dual.

The hierarchy of linear program formulations, [LP₁], [LP₂], and [LP₃], all retain the set of integer allocations but prune additional fractional solutions. Each formulation introduces new constraints into the primal, with the dual problems [DLP₁], [DLP₂], and [DLP₃] containing richer price structures. For example, in [DLP₁] the prices on a bundle are linear in the price of items, i.e. $p(S) = \sum_{j \in S} p(j)$, where $p(j)$ is the price of item j in bundle S . Moving to [DLP₂], the price on a bundle can be non-linear in the price on items, and in [DLP₃] the price on a bundle can be different to different agents. Bikchandani & Ostroy prove that LP₃ solves all CAP instances, and demonstrate the existence of competitive equilibrium prices, even though they must sometimes be both non-linear and non-anonymous.

Solving the CAP with the high-level linear program formulations is likely to be less efficient computationally than direct search-based methods applied to the integer program formulation. Formulations [LP₂] and [LP₃] introduce an exponential number of additional primal constraints, and dual variables, effectively enumerating *all possible* solutions to the CAP. In comparison, search methods, such as branch-and-bound with LP-based heuristics, solve the problem with implicit enumeration and pruning.

However the formulations are very useful in the context of mechanism design and decentralized CAP problems. In Section 4.6 I present COMBAUCTION, a primal-dual algorithm for the CAP, which

(a) computes optimal primal and dual solutions *without* complete information about agent valuation functions.

(b) computes optimal primal and dual solutions *without* complete enumeration of all primal constraints and/or dual variables.

In fact most of the computation within COMBAUCTION occurs in winner determination, which solves the restricted primal problem in each round, and winner-determination itself is solved with a branch-and-bound search algorithm.

4.4.1 Integer Program Formulation

Introducing $x_i(S)$ to indicate that agent i receives bundle S the straightforward integer program, [IP], formulation of the combinatorial allocation problem is:

$$\max_{x_i(S)} \sum_S \sum_i x_i(S) v_i(S) \quad [\text{IP}]$$

$$\text{s.t. } \sum_S x_i(S) \leq 1, \quad \forall i \quad (\text{IP-1})$$

$$\sum_{S \ni j} \sum_i x_i(S) \leq 1, \quad \forall j \quad (\text{IP-2})$$

$$x_i(S) \in \{0, 1\}, \quad \forall i, S$$

where $S \ni j$ indicates a bundle S that contains item j . The objective is to compute the allocation that maximizes value over all agents, without allocating more than one bundle to any agent (IP-1) and without allocating a single item multiple times (IP-2). Let V_{IP}^* denote the value of the optimal allocation.

4.4.2 First-order LP Formulation

LP_1 is a direct linear relaxation, which replaces the integral constraints $x_i(S) \in \{0, 1\}$ with non-negativity constraints, $x_i(S) \geq 0$.

$$\max_{x_i(S)} \sum_S \sum_i x_i(S) v_i(S) \quad [\text{LP}_1]$$

$$\text{s.t. } \sum_S x_i(S) \leq 1, \quad \forall i \quad (\text{LP}_1\text{-1})$$

$$\sum_{S \ni j} \sum_i x_i(S) \leq 1, \quad \forall j \quad (\text{LP}_1\text{-2})$$

$$x_i(S) \geq 0, \quad \forall i, S$$

$$\min_{p(i), p(j)} \sum_i p(i) + \sum_j p(j) \quad [\text{DLP}_1]$$

$$\text{s.t. } p(i) + \sum_{j \in S} p(j) \geq v_i(S), \quad \forall i, S \quad (\text{DLP}_1\text{-1})$$

$$p(i), p(j) \geq 0, \quad \forall i, j$$

Prices $p(j)$ on items $j \in \mathcal{G}$ define a feasible dual solution, with the substitution $p(i) = \max_S \{v_i(S) - \sum_{j \in S} p(j)\}$.

PROPOSITION 4.2 (first-order dual). *The value of the first-order dual is the sum of the maximal utility to each agent plus the total price over all items (this is the auctioneer's maximal revenue).*

	A	B	AB
Agent 1	0	0	3
Agent 2	2*	0	2
Agent 3	0	2*	2

Table 4.1: Problem 1.

	A	B	C	AB	BC	AC	ABC
Agent 1	60	50	50	200*	100	110	250
Agent 2	50	60	50	110	200	100	255
Agent 3	50	50	75*	100	125	200	250

Table 4.2: Problem 2.

The dual variables define linear prices, the price for bundle $S \subseteq \mathcal{G}$ is $p(S) = \sum_{j \in S} p(j)$. From Definition 4.7 the optimal dual solution defines competitive equilibrium prices if and only if a partition of items exists at the prices that allocates each agent a bundle in its utility-maximizing set and allocates every item with positive price exactly once.

Problem 1 in Table 4.1 can be solved with $[\text{LP}_1]$; $V_{\text{LP}_1}^* = V_{\text{IP}} = 4$. The optimal allocation is $x_2(A) = 1$ and $x_3(B) = 1$, indicated by *. To see that $V_{\text{LP}_1} \leq 4$, notice that dual prices $p(A) = p(B) = 1.6$ gives a dual solution with value $V_{\text{DLP}_1} = 0 + 0.4 + 0.4 + 3.2 = 4$. Remember that $V_{\text{LP}_1}^* \leq V_{\text{DLP}_1}$ for all dual solutions by the weak-duality theorem of linear programming. These are one set of competitive equilibrium prices.

However, in general the value $V_{\text{LP}_1}^* > V_{\text{IP}}^*$ and the optimal primal solution makes fractional assignments to agents. As an example of when $[\text{LP}_1]$ fails, consider Problem 2 in Table 4.2. In this problem $V_{\text{LP}_1}^* = 300 > V_{\text{IP}}^* = 275$. The primal allocates fractional solution $x_1(AB) = 0.5$, $x_2(BC) = 0.5$ and $x_3(AC) = 0.5$, which satisfies constraints (LP₁-1) because $\sum S \ni j \sum_i x_i(S) \leq 1$ for all items $j \in \mathcal{G}$. Prices $p(A) = p(B) = p(C) = 100$ solve the dual problem DLP_1 .

Kelso & Crawford [KC82] prove that gross-substitutes (GS) preferences are a sufficient condition for the existence of linear competitive equilibrium prices, such that $V_{\text{LP}_1}^* = V_{\text{IP}}^*$.

To define gross-substitutes preferences, let $D_i(p)$ define the demand set of agent i at prices p , i.e. the set of bundles that maximize its utility (value - price).

DEFINITION 4.8 [gross-substitutes (GS)] For all price vectors p, p' such that $p' \geq p$, and all $S \in D_i(p)$, there exists $T \in D_i(p')$ such that $\{j \in S : p_j = p'_j\} \subset T$.

In words, an agent has GS preferences if an agent continues to demand items with the same price as the price on other items increases. If preferences are also *monotonic*, such that $v_i(S') \geq v_i(S)$ for all $S' \supseteq S$, then GS implies *submodular* preferences.

DEFINITION 4.9 [submodular preferences] Valuation function $v_i(S)$ is submodular if for all $S, T \subseteq \mathcal{G}$,

$$v_i(S) + v_i(T) \geq v_i(S \cup T) + v_i(S \cap T)$$

Submodularity is equivalent to a generalized statement of *decreasing returns*:

DEFINITION 4.10 [decreasing returns] Valuation function $v_i(S)$ has decreasing marginal returns if for all $S \subset T \subseteq \mathcal{G}$ and all $j \in \mathcal{G}$,

$$v_i(T) - v_i(T \setminus \{j\}) \leq v_i(S) - v_i(S \setminus \{j\})$$

In other words, the value of an item increases as it is introduced to larger sets of items. Subadditivity implies that the value for any bundle is no greater than the minimal sum of values for a partition of the bundle.

In fact, gross-substitutes preferences define the largest set of preferences that contain unit-demand preferences (see Definition 4.14) for which the existence of linear competitive equilibrium prices can be shown [GS99].

The rest of this section introduces two alternative linear program formulations of CAP, [LP₂] and [LP₃], due to Bikchandani & Ostroy [BO99].

4.4.3 Second-order LP Formulation

Introducing new constraints to the first-order linear program relaxation [LP₁] of [IP] gives a second-order linear program [LP₂] with dual [DLP₂]. The corresponding dual variables to the new primal constraints are interpreted as *bundle prices* within an auction-based primal-dual algorithm.

$$\begin{aligned}
& \max_{x_i(S), y(k)} \sum_S \sum_i x_i(S) v_i(S) && \text{[LP}_2\text{]} \\
\text{s.t. } & \sum_S x_i(S) \leq 1, \quad \forall i && \text{(LP}_2\text{-1)} \\
& \sum_i x_i(S) \leq \sum_{k \ni S} y(k), \quad \forall S && \text{(LP}_2\text{-2)} \\
& \sum_k y(k) \leq 1 && \text{(LP}_2\text{-3)} \\
& x_i(S), y(k) \geq 0, \quad \forall i, S, k
\end{aligned}$$

$$\begin{aligned}
& \min_{p(i), p(S), \pi} \sum_i p(i) + \pi && \text{[DLP}_2\text{]} \\
\text{s.t. } & p(i) + p(S) \geq v_i(S), \quad \forall i, S && \text{(DLP}_2\text{-1)} \\
& \pi - \sum_{S \in k} p(S) \geq 0, \quad \forall k && \text{(DLP}_2\text{-2)} \\
& p(i), p(S), \pi \geq 0, \quad \forall i, S
\end{aligned}$$

where $k \in K$ is a *partition* of items in set K , and $k \ni S$ indicates that bundle S is represented in partition k . A partition is a feasible “bundling” of items, e.g. $[A, B, C]$ or $[AB, C]$, etc., and K is the set of all possible partitions, e.g. $K = \{[A, B, C], [AB, C], [A, BC], \dots, [ABC]\}$ in Problem 2 (Table 4.2).

Constraints (LP₂-2) and (LP₂-3) replace constraints (LP₁-1), and ensure that no more than one unit of every item is allocated. The dual [DLP₂] has variables $p(i)$, $p(S)$ and π , which correspond to constraints (LP₂-1), (LP₂-2) and (LP₂-3), and constraints (DLP₂-1) and (DLP₂-2) correspond to primal variables $x_i(S)$ and $y(k)$.

Dual variables $p(S)$ can be interpreted as bundle prices, and with substitution $p(i) = \max_S \{v_i(S) - p(S)\}$, i.e. the maximal utility to agent i at prices $p(S)$, and $\pi = \max_{k \in K} \sum_{S \in k} p(S)$, i.e. the maximal revenue to the auctioneer at prices $p(S)$.

PROPOSITION 4.3 (second-order dual). *The value of the dual is the sum of the maximal utility to each agent with bundle prices $p(S)$ plus the auctioneer’s maximal revenue over all feasible (and non-fractional) allocations at the prices.*

	A	B	AB
Agent 1	0	0	3^*
Agent 2	2	2	2

Table 4.3: Problem 3.

The dual variables correspond to bundle prices, $p(S)$, and the optimal dual solution defines competitive equilibrium prices (by Definition 4.7) if there is an allocation that gives each agent a bundle in its utility-maximizing set at the prices, and maximizes revenue to the auctioneer over all possible allocations.

With the additional constraints $[\text{LP}_2]$ solves Problem 2. Allocation $x_1(AB) = x_2(BC) = x_3(AC) = 0.5$ is *not* feasible in $[\text{LP}_2]$ because it is not possible to allocate $y(k_1) = y(k_2) = y(k_3) = 0.5$ for $k_1 = [AB, C]$, $k_2 = [AC, B]$ and $k_3 = [AB, C]$ without violating constraint (LP₂-3) and without this we violate constraints (LP₂-2). $[\text{LP}_2]$ solves Problem 2, with $V_{\text{LP}_2}^* = V_{\text{IP}}^* = 275$. An optimal dual solution is given by bundle prices $p = (50, 60, 75, 190, 200, 200, 255)$, with total agent maximal utility $10 + 0 + 0$ and maximal auctioneer revenue $75 + 190 = 265$, i.e. $V_{\text{DLP}_2}^* = 275$.

However, Problem 3 is an example that $[\text{LP}_2]$ does not solve. The value of the optimal primal solution is $V_{\text{LP}_2}^* = 3.5$, which is greater than the value of the optimal feasible allocation, $V_{\text{IP}}^* = 3$. The primal allocates fractional bundles $x_1(AB) = 0.5$ and $x_2(A) = x_2(B) = 0.5$, which satisfies constraints (LP₂-2) and (LP₂-3) with $y(k_1) = y(k_2) = 0.5$ for partitions $k_1 = [AB, \emptyset]$ and $k_2 = [A, B]$. Prices $p(A) = 1.5, p(B) = 1.5, p(AB) = 3$ solves the dual problem DLP_2 .

4.4.4 Third-order LP Formulation

Introducing new constraints to the second-order linear program relaxation $[\text{LP}_2]$ of $[\text{IP}]$ gives a third-order linear program $[\text{LP}_3]$ with dual $[\text{DLP}_3]$. The corresponding dual variables to the new primal constraints are interpreted as *non-anonymous*, or *discriminatory* bundle prices, with different prices for the same bundle to different agents.

$$\begin{aligned}
& \max_{x_i(S), y(k)} \sum_S \sum_i x_i(S) v_i(S) && [\text{LP}_3] \\
\text{s.t.} \quad & \sum_S x_i(S) \leq 1, \quad \forall i && (\text{LP}_3\text{-1}) \\
& x_i(S) \leq \sum_{k \ni [i, S]} y(k), \quad \forall i, S && (\text{LP}_3\text{-2}) \\
& \sum_k y(k) \leq 1 && (\text{LP}_3\text{-3}) \\
& x_i(S), y(k) \geq 0, \quad \forall i, S, k
\end{aligned}$$

$$\begin{aligned}
& \min_{p(i), p_i(S), \pi} \sum_i p(i) + \pi && [\text{DLP}_3] \\
\text{s.t.} \quad & p(i) + p_i(S) \geq v_i(S), \quad \forall i, S && (\text{DLP}_3\text{-1}) \\
& \pi - \sum_{[i, S] \in k} p_i(S) \geq 0, \quad \forall k && (\text{DLP}_3\text{-2}) \\
& p(i), p_i(S), \pi \geq 0, \quad \forall i, S
\end{aligned}$$

where $k \ni [i, S]$ indicates that *agent-partition* $k \in K'$ contains bundle S designated for agent i . Variable $y(k)$ in $[\text{LP}_3]$ corresponds to an *agent-partition* k , where the set of agent-partitions in Problem 3 is $K' = \{[(1, A), (2, B)], [(1, B), (2, A)], [(1, AB), (2, \emptyset)], [(1, \emptyset), (2, AB)]\}$. It is important to note that each agent can receive at most one bundle in a particular agent-partition.

The dual variables $p_i(S)$ that correspond to primal constraints (LP₃-2) are interpreted as *non-anonymous* bundle prices, price $p_i(S)$ is the price to agent i for bundle S . As before, substitutions $p(i) = \max_S \{v_i(S) - p_i(S)\}$, i.e. the maximal utility to agent i at individual prices $p_i(S)$, and $\pi = \max_{k \in K'} \sum_{[i, S] \in k} p_i(S)$, i.e. the maximal revenue to the auctioneer at prices $p_i(S)$ given that it can allocate at most one bundle at prices $p_i(S)$ to each agent i .

PROPOSITION 4.4 (third-order dual). *The value of the dual to $[\text{LP}_3]$ is the sum of the maximal utility to each agent with bundle prices $p_i(S)$ plus the auctioneer's maximal revenue over all feasible allocations at the prices. In this case an allocation is feasible if it allocates no more than one bundle to each agent.*

The dual variables correspond to non-anonymous bundle prices, $p_i(S)$, and the optimal dual solution defines competitive equilibrium prices if there is an allocation of items that simultaneously gives each agent a bundle in its utility-maximizing set and maximizes the auctioneer’s revenue, over all possible allocations that sell at most one bundle to each agent.

Bikchandani & Ostroy [BO99] prove this important theorem:

THEOREM 4.5 (integrality). *The optimal solution to linear program $[LP_3]$ is always integral, and therefore an optimal solution to CAP, with $V_{LP_3}^* = V_{DLP_3}^* = V_{IP}^*$.*

Therefore, there are always competitive equilibrium bundles prices for CAP, although these prices must be non-anonymous in some problems.

Consider Problem 3. Allocation $x_1(AB) = 0.5$ and $x_2(A) = x_3(B) = 0.5$ is *not* feasible in $[LP_3]$ because $y(k_1) = y(k_2) = y(k_3) = 0.5$ for $k_1 = [(1, AB), (2, \emptyset)]$, $k_2 = [(1, A), (2, B)]$ and $k_3 = [(1, B), (2, A)]$ violates constraint (LP₃-3), but without this constraints (LP₃-2) are violated. In this problem $V_{LP_3}^* = V_{IP}^* = 3$. To see this, consider bundle prices $p_1 = (0, 0, 2.5)$ and $p_2 = (2, 2, 2)$, for which the value of the dual is $0.5 + 0 + 2.5 = 3$. This proves that $V_{LP_3} \leq 3$ by the weak-duality theorem of linear programming.

I will return to this hierarchy of linear-program formulations of the CAP in Section 4.6, when I introduce the COMBAUCTION primal-dual algorithm. COMBAUCTION constructs feasible primal and dual solutions to an appropriate linear program formulation, and adjusts the solution until complementary-slackness conditions are also satisfied.

4.5 Tractable Combinatorial Allocation Problems

The CAP is equivalent to the maximum weighted set packing problem (SPP), a well-studied problem in the operations research literature. In SPP there are a set of items, and a set of subsets each with non-negative weights, and the goal is to pack the items into sets to maximize total value, without using any item more than once. CAP can be reduced to SPP by introducing an additional “dummy item” for the XOR bids from each agent. de Vries & Vohra [dVV00] also note two closely related problems, the set partitioning problem (SPA), in which the goal is to select a set of subsets with minimal

cost that include all items at most once, and the set covering problem (SCP), in which the goal is to select a set of subsets with minimal cost that include all items at least once. Set covering problems find applications in railway crew-scheduling and airline scheduling, where items are flights/trains, and bundles represent possibility sets for individual workers. A considerable amount is known about the complexity of this class of problems.

A classic technique in combinatorial optimization theory is to relax an integer program to a linear one. Many tractable special cases follow by considering the conditions on the natural relaxation of the integer program that provide integer solutions. For example, one sufficient condition is that the linear program is *integral*, such that all extremal feasible points are integral, i.e. 0-1. In this case the integrality requirement can be dropped and the problem solved as a linear program in polynomial time. Restrictions on the constraint matrix, corresponding to restrictions on the kinds of subsets permitted in CAP, can provide this integrality property [dVV00].

Additional restrictions, for example on the size of bids, or on the valuation structure of bids, can also lead to tractable special cases. Given the connection with linear programming relaxations this is a good place to review known tractable special-cases in the literature. The results here are drawn from Rothkopf *et al.* [RPH98], de Vries & Vohra [dVV00], Nisan [Nis00], and earlier work due to Kelso & Crawford [KC82].

It is important to understand the characteristics of tractable special-cases of CAP because this knowledge can be leveraged within mechanism design, achieving tractable and strategy-proof solutions (see Section 3.2.1 in Chapter 3).

Restrictions on Structure of Bundles

Table 4.4 presents tractable instances of CAP that follow from restrictions on the types of bundles on which agents can submit bids. de Vries & Vohra note that the linear-ordering (or consecutive ones) condition implies that the constraint matrix satisfies *total unimodularity*,¹ and that the nested-hierarchical structure implies that the constraint matrix is *balanced*.² Nisan [Nis00] provides a proof-by-induction that the linear program has integral solutions in these cases, and also describes a method to combine two bid structures with the integral property into a single structure that retains the property.

¹A matrix satisfies total unimodularity if the determinant of every square submatrix is 0, 1, or -1.

²A 0-1 matrix is balanced if it has no square submatrix of odd order with exactly two 1's in each row and column.

linear-order	ordering $G = (g_1, g_2, \dots, g_n)$	[RPH98]
circular ones	every bid is for a contiguous sequence	
nested-hierarchical	also allow bids of form $g_n g_1 g_2$, etc.	[RPH98]
	for every two subsets of items S_1, S_2	[RPH98]
	that appear as part of any bid they are either disjoint or one contains the other	
or-singletons	bids for single-items	-
single-item bids	one item	-
bids for pairs of items	cardinality constraint on size of bids	[RPH98]
multi-unit, decreasing returns	identical items, each agent has decreasing value for each additional item	[Nis00]

Table 4.4: Tractable structure on bids

non-decreasing and supermodular	“increasing returns”	[dVV00]
two-types of agents		
gross-substitutes	“decreasing-returns”	[KC82]
unit-demand	agents only want one item	[Kuh55]
linear-additive	agents have linear values across items	[CK81]

Table 4.5: Constraints on valuation functions

Restrictions on Values on Bundles

Table 4.5 presents tractable instances of CAP that follow from restrictions on the value structure of agents bids. de Vries & Vohra [dVV00] note that the non-decreasing and supermodular preferences condition again provides the linear program relaxation of the CAP with integral solutions. Gross-substitutes were defined earlier in Definition 4.8 and have an intuitive interpretation as decreasing-returns, and also imply submodular preferences.

DEFINITION 4.11 [supermodular preferences] Bid function $b_i(S)$ is supermodular if for all $S, T \subseteq \mathcal{G}$,

$$b_i(S) + b_i(T) \leq v_i(S \cup T) + v_i(S \cap T)$$

The equivalence of supermodularity and increasing returns is well-known in the literature [GS99].

DEFINITION 4.12 [increasing returns] Bid function $b_i(S)$ has increasing marginal returns if for all $S \subset T \subseteq \mathcal{G}$ and all $j \in \mathcal{G}$,

$$b_i(T) - v_i(T \setminus \{j\}) \geq b_i(S) - v(S \setminus \{j\})$$

Note carefully that we can have any number of different types of submodular valuation functions, one from each agent, but only at most *two* different types of supermodular functions if the CAP problem is to be tractable. It is easier to solve a maximization problem, such as the CAP, with submodular (convex) objective functions than supermodular (concave) objective functions.

Exact Solutions

Rothkopf *et al.* [RPH98] also suggest a dynamic programming algorithm for CAP, which has run-time complexity independent of the number of bids actually placed, but quickly becomes intractable for large numbers of items, with scaling property $O(3^m)$ in the number of items m . Branch-and-bound search methods, either with AI-based heuristics [San99, FLBS99], or with linear-program based heuristics [ATY00] have also been studied for general CAP instances.

Approximate Solutions

The CAP is difficult to approximate, at least within a worst-case multiplicative factor. There is no polynomial time algorithm with a reasonable worst-case guarantee [Has99].

Approximation algorithms in the literature without this guarantee include a local-search approach [HB00], a simple “relax and round” method [Nis00], and iterative methods [FLBS99]. COMBAUCTION can itself be viewed as an approximate algorithm for CAP. COMBAUCTION provides a worst-case bound on the difference between the value of its solution and the value of the optimal solution. This error-term increases linearly with the minimal bid increment, which defines the rate at which prices are increased across rounds, while the number of rounds in the auction is inversely-proportional to the minimal bid increment. A larger bid increment reduces the number of rounds in the auction, reducing the number of winner-determination problems the auction must solve, in return for a loss in worst-case efficiency. Experimental results in Section 5.5.1 show the effectiveness of this approach.

4.6 COMBAUCTION: A Primal-Dual Method for CAP

COMBAUCTION is a primal-dual algorithm for the linear program models of CAP introduced in Section 4.4. The algorithm terminates with optimal primal and dual solutions to an appropriate level in the linear-program hierarchy, selecting the price structure dynamically to support the optimal allocation in equilibrium. In the next chapter I describe the *i*Bundle auction, which is an auction-based implementation of COMBAUCTION for agents that follow myopic best-response bidding strategies.

In COMBAUCTION the prices are linear and anonymous in special cases, non-linear and anonymous (bundle prices) in many problems, and non-linear and non-anonymous when that is necessary to strengthen the dual formulation of the CAP. The decision about anonymous vs. non-anonymous pricing is made dynamically during the algorithm, while non-linear prices are introduced whenever an agent bids for a bundle of items instead of individual items.

4.6.1 Description

Let $p_{\text{ask}}(S) \geq 0$ denote anonymous ask prices on bundles $S \subseteq \mathcal{G}$. Set the initial ask prices to zero, and use only anonymous prices at the start of the auction. The prices represent a feasible *dual solution* in each round of the algorithm. Non-anonymous prices are denoted $p_{\text{ask},i}(S)$, for the price on bundle S to agent i , and initially $p_{\text{ask},i}(S) = p_{\text{ask}}(S)$ for all agents and all bundles.

Let $\hat{S} = (\hat{S}_1, \dots, \hat{S}_I)$ denote the provisional allocation, computed from agents' bids in each round to maximize revenue, where agent i is provisionally allocated bundle \hat{S}_i . Let $p_i^+(S)$ denote *personalized* ask prices to agent i , computed as follows:

$$p_i^+(S) = \begin{cases} p_{\text{ask},i}(S) & , \text{ if } p_{\text{ask},i}(S) \leq v_i(S) \text{ and } S \neq \hat{S}_i \\ p_{\text{ask},i}(S) - \epsilon & , \text{ otherwise} \end{cases}$$

The personalized ask price is the price that agent i can actually bid for bundle S . In words, an agent must bid the ask price unless the ask price is greater than its value or the agent receives the bundle in the current allocation, in which case it can bid ϵ below the ask price.

Figure 4.5 describes COMBAUCTION. The basic variation, also known as COMBAUCTION(D), introduces non-anonymous prices whenever agent bids are not “safe”. Two other

```

COMBAUCTION
input: agent values  $v_i(\cdot)$ 
stop = false;  $\hat{S} = \emptyset$ ;  $\mathbf{p}_{\text{ask}}(\cdot) = 0$ ; anon =  $\mathcal{I}$ ;
while ( $\neg$  stop) {
  update personalized prices  $p_i^+(S)$  for every agent  $i$ ;
  compute best-response set  $\text{BR}_i(\mathbf{p}_i^+)$  for every agent  $i$ ;
  compute partition of items  $\hat{S} = (\hat{S}_1, \dots, \hat{S}_I)$  to maximize revenue,
  subject to  $\hat{S}_i \in \text{BR}_i(\mathbf{p}_i^+)$ .;
  if ( ( $\text{BR}_i(\mathbf{p}_i^+) = \emptyset$ ) or unchanged( $\text{BR}_i(\mathbf{p}_i^+)$ ) or ( $\hat{S}_i \neq \emptyset$ ) ) for every  $i$ 
    stop = true;
  else {
    for every  $j$  with (( $\text{BR}_j(\mathbf{p}_j^+) \neq \emptyset$ ) and ( $\hat{S}_j = \emptyset$ )) {
      if ( (safe( $\text{BR}_j(\mathbf{p}_j^+)$ )) and (  $j \in \text{anon}$ ))
         $\mathbf{p}_{\text{ask}} = \text{anon\_update}(\text{BR}_j(\mathbf{p}_j^+), \mathbf{p}_{\text{ask}})$ ;
      else {
         $\mathbf{p}_{\text{ask},j} = \text{nonanon\_update}(\text{BR}_j(\mathbf{p}_j^+), \mathbf{p}_{\text{ask},j})$ ;
         $\text{anon} = \text{anon} \setminus \{j\}$  ;
      }
    }
  }
}
output: final allocation  $\hat{S}$ , final prices  $\mathbf{p}_{\text{ask},i}(\hat{S}_i)$ .

```

Figure 4.5: The COMBAUCTION algorithm. Special cases: COMBAUCTION(2) sets *safe*($\text{BR}_j(\mathbf{p}_j^+)$) true in every iteration; COMBAUCTION(3) sets *safe*($\text{BR}_j(\mathbf{p}_j^+)$) false in every iteration.

important variations include COMBAUCTION(2), in which the safe condition is always assumed **true**, and COMBAUCTION(3), in which the safe condition is always assumed **false**. Labels (D), (2), and (3) correspond to “dynamic” non-anonymous pricing, second-order pricing (i.e. non-linear), and third-order pricing (i.e. non-linear and non-anonymous).

Prices are initially zero on all bundles, and identical across all agents such that $p_{\text{ask},i}(S) = p_{\text{ask}}(S)$ for all i and all S . At the start of each iteration each the new personalized prices \mathbf{p}_i^+ are computed for each agent. Next, each agent $i \in \mathcal{I}$ reports its best-response set $\text{BR}_i(\mathbf{p}_i^+)$. The best-response set is the set of all bundles and corresponding personalized prices that maximize utility to within $\epsilon > 0$:

$$\text{BR}_i(\mathbf{p}_i^+) = \{(S, p_i^+(S)) \mid v_i(S) - p_i^+(S) + \epsilon \geq \max(u_i^*(\mathbf{p}_i^+), 0)\}$$

where $u_i^*(\mathbf{p}_i^+) = \max_S v_i(S) - p_i^+(S)$, i.e. the maximal utility over all bundles at the current (personalized) prices. Constant $\epsilon > 0$ is the minimal bid increment, and controls

the rate at which prices are increased across rounds.

The provisional allocation $\hat{\mathbf{S}} = (\hat{S}_1, \dots, \hat{S}_I)$, is computed from agents' bids to maximize revenue:

$$\begin{aligned} & \max_{(S_1, \dots, S_I)} \sum_{i \in \mathcal{I}} p_i^+(S_i) && \text{(WD problem)} \\ \text{s.t.} & && (S_i, p_i^+(S_i)) \in \text{BR}_i(\mathbf{p}_i^+) \\ & && S_i \cap S_j = \emptyset, \quad \forall i, j \end{aligned}$$

Prices are increased based on bids from agents not in the current provisional allocation. In each round the agents $i \in anon$ receive anonymous prices, $p_{ask}(S)$, and the agents $i \notin anon$ receive non-anonymous prices, $p_{ask,i}(S)$, which can charge a different price for the same bundle to different agents. Initially every agent receives anonymous prices, and $anon = \mathcal{I}$.

The price-update step depends on whether an agent's bids are *safe*. The *safe* condition is defined on a set of bundles \mathcal{S} as follows:

$$safe(\mathcal{S}) = \neg disjoint(\mathcal{S}) \text{ or } (|\mathcal{S}| = 1)$$

where $disjoint(\mathcal{S})$ is true if there is at least one pair of bundles $S, T \in \mathcal{S}$ that are non-overlapping, such that $S \cap T = \emptyset$, and $|\mathcal{S}| = 1$ denotes a best-response set with only one bundle.

Price increases based on unsuccessful but *safe* bids from an agent j in $anon$ are computed with update rule `anon_update`($\text{BR}_j(\mathbf{p}_j^+)$, \mathbf{p}_{ask}), which increases the price $p_{ask}(S)$ to $p_j^+(S) + \epsilon$ on all bundles S in agent j 's best-response bid set, where constant $\epsilon > 0$ is the *minimal bid increment*. This price increase affects all agents in $anon$. Notice that if the agent's personalized bid price is ϵ below the current ask price, for example if an agent is repeating a bid for a bundle in the provisional allocation, then the price on this unsuccessful bid does *not* increase the price on the bundle.

In the first round that an agent's bids are unsuccessful and fail the *safe* condition the agent is removed from the anonymous set and faces individual prices in all future rounds. Price increases to an agent not in $anon$ are based only on its own bids, and its bids never directly affect the prices to agents that remain in $anon$. Individual ask prices are initially set to the current anonymous prices. Price update rule `nonanon_update`($\text{BR}_j(\mathbf{p}_j^+)$, $\mathbf{p}_{ask,j}$)

increases the price $p_{\text{ask},j}(S)$ to agent j to $p_j^+(S) + \epsilon$, for every bundle S in its best-response set. Again, this will only *increase* the price if the personalized price is not discounted by ϵ from the ask price.

COMBAUCTION terminates when every agent with a non-empty best-response set *either* receives a bundle in the provisional allocation:

$$\text{BR}_i(\mathbf{p}_i^+) \neq \emptyset \quad \Rightarrow \quad \hat{S}_i \neq \emptyset \quad \text{condition [T1]}$$

or submits the same best-response bids in two successive rounds (condition [T2]).

COMBAUCTION(2) is the special-case of COMBAUCTION for which the safety condition is assumed true in all rounds and all agents remain in the anonymous set. COMBAUCTION(D) is the regular version of COMBAUCTION, described above, in which non-anonymous prices are introduced dynamically. COMBAUCTION(3) is a variation in which the safety condition is assumed false in all rounds and all agents face individual prices in every round, i.e. the set $\text{anon} = \emptyset$ from round 1.

Discussion

The personalized prices ensure that price increases do not “overshoot” the values of agents that receive a bundle in the efficient allocation. Suppose for example that the allocation problem has a single item, and there are two agents with the same value for the item. It is essential that when the agents eventually drive the price above their value, one of the agents, i.e. the agent with the item in the final provisional allocation, is able to repeat a bid for the item at the previous price— just before it was forced to high.

Allowing agents to submit a best-response bid set, that can include more than one bundle, and is accurate to within ϵ , is also important in solving some problems. Consider for example that agents 1 and 2 both want A or B , and a third agent that wants AB . Allowing agents 1 and 2 to bid for both A and B , but only receive one of A or B in the allocation, solves the coordination problem, in which agents 1 and 2 must be allocated different items to out-bid the third agent.

4.6.2 Optimality Result

We first state an optimality result for the variations on COMBAUCTION with non-anonymous prices.

(CS1a)	Agents maximize utility: $S_i^* \neq \emptyset \Rightarrow S_i^* = \arg \max_{S_i} v_i(S_i) - p_i(S_i)$ <i>true in every round because of best-response</i>
(CS1b)	Agents not in allocation happy: $S_i^* = \emptyset \Rightarrow \max_{S_i} v_i(S_i) - p_i(S_i) \leq 0$ <i>true in final round because of termination condition</i>
(CS2)	Auctioneer maximizes revenue: $S^* = \arg \max_{(S_1, \dots, S_I)} \sum p_i(S_i)$ <i>true in every round because of price-update rules</i>

Table 4.6: Proof outline for COMBAUCTION

THEOREM 4.6 (COMBAUCTION optimality). *(Optimality) COMBAUCTION(D) and COMBAUCTION(3) compute an allocation with value within $3 \min\{|\mathcal{G}|, |\mathcal{I}|\}\epsilon$ of optimal.*

for $|\mathcal{G}|$ items, $|\mathcal{I}|$ agents, and ϵ bid increment.

COROLLARY 4.1 COMBAUCTION(D) and COMBAUCTION(3) compute the efficient allocation and competitive equilibrium prices for a small enough bid increment ϵ .

Clearly as ϵ gets smaller than the smallest finite difference in agents' values for bundles this converges to the optimal solution. Table 4.6 provides a sketched proof, while a full proof is provided below.

Recall that within COMBAUCTION it is assumed that agents provide best-response information in response to prices in each iteration. Corresponding statements of the efficiency of *iBundle*, the auction interpretation of COMBAUCTION, make assumptions about agent behavior explicit (see Chapter 5).

Algorithm COMBAUCTION(2), without the safety check and without non-anonymous prices, is also provably optimal in the following (sufficient) conditions:

THEOREM 4.7 (anonymous optimality). *COMBAUCTION(2) is an optimal primal-dual algorithm for CAP with anonymous prices in the following special-cases:*

(a) agents have additive or superadditive values, i.e. $v(S \cup S') \geq v(S) + v(S')$ for non-conflicting bundles S and S' ; (b) agents demand bundles from the same partition of items,

e.g. all bids are for pairs of matching shoes, or single items; (c) the demand set of bundles for agent i , the bundles it bids for over the auction, is disjoint from the demand set of agent j , for all agents $i \neq j$; (d) bids from each agent are always overlapping; (e) bids from each agent are always for a single bundle.

A proof of conditions (a–c) follow quite easily from the proof of the optimality of COMBAUCTION(D). When these conditions hold the auctioneer is sure to maximize revenue from bids in the next round of the auction without introducing non-anonymous prices, even in the case that bids from agents break the “safety” condition.³ In case (b) the auction reduces to a simultaneous ascending-price auction on bundles in a fixed partition of items. The assignment problem, in which agents have unit-demand for items, is a special case of (b).

Conditions (d–e) follow trivially from the main result, because the safety condition holds in all rounds under these conditions. Single-minded bidders [LOS99] satisfy (e). Bidders that demand a core set of items with a selection from an additional set of items satisfy (d); consider for example a bidder in the FCC spectrum auction that needs New York, and then would like as many of the geographically neighboring licenses as possible.

4.6.3 Proof: COMBAUCTION(2)

The optimality proof of COMBAUCTION is inspired by a proof due to Bertsekas [Ber87] for AUCTION, an iterative primal-dual algorithm with an auction interpretation for the assignment problem.

I first prove optimality for COMBAUCTION(2), in the special-case that the best-response bid sets are safe in all rounds of the auction. The proof of optimality for COMBAUCTION(D) and COMBAUCTION(3) follows from an equivalence between COMBAUCTION(2) with a dummy item introduced for each agent and appended to its bids, and COMBAUCTION(3).

In outline, I show that COMBAUCTION implements a primal-dual algorithm for [LP₂] and [DLP₂], and computes integral solutions to [LP₂] when agents follow myopic best-response bidding strategies and bids are safe.

³For example, in the case of superadditive values whenever an agent bids for a compatible pair of bundles S and S' that violate the safety condition it must be the case that $p(S \cup S') > p(S) + p(S')$. This condition is sufficient to show that the auctioneer can continue to maintain (CS2) and maximize revenue from agent bids in the next round.

First, I show that the allocation and prices in each round of the auction correspond to feasible primal and dual solutions. Then, I show that the primal and dual solutions satisfy complementary-slackness conditions when the auction terminates.

In a particular round, let \hat{S}_i denote the provisional allocation to agent i , and $p_{\text{ask}}(S)$ denote the ask price for bundle S .

Feasible primal. To construct a feasible primal solution assign $x_i(\hat{S}_i) = 1$ and $x_i(S') = 0$ for all $S' \neq \hat{S}_i$. Partition $y(k^*) = 1$ for $k^* = [\hat{S}_1, \dots, \hat{S}_{|I|}]$, and $y(k) = 0$ otherwise.

Feasible dual. To construct a feasible dual solution let dual variable $p(S)$ equal the ask price, $p_{\text{ask}}(S)$, on bundle S in COMBAUCTION. The following values for $p(i)$ and π then satisfy constraints (DLP-1) and (DLP-2):

$$p(i) = \max \left\{ 0, \max_{S \subseteq G} \{v_i(S) - p(S)\} \right\} \quad (4.1)$$

$$\pi = \max_{k \in K} \sum_{S \in k} p(S) \quad (4.2)$$

The value $p(i)$ can be interpreted as agent i 's maximum utility at the ask prices, and π can be interpreted as the maximum revenue that the auctioneer can achieve at the ask prices (irrespective of the bids placed by agents).

It is not necessary to explicitly compute $p(i)$, instead we will prove that allocation \hat{S} and prices $p_{\text{ask}}(S)$ correspond to primal and dual solutions that satisfy complementary slackness conditions when the auction terminates.⁴

Complementary-slackness conditions. The first primal CS condition, CS-1 is:

$$x_i(S) > 0 \Rightarrow p(i) + p(S) = v_i(S) \quad (\text{CS-1})$$

Given (4.1) it states that all agents must only receive a bundle that maximizes utility at the current prices. CS-1 is maintained throughout the auction because bundles are only allocated according to bids from agents, and agents place best-response bids.

Based on the best-response bidding strategy, we have

$$v_i(\hat{S}_i) - p_i^+(S) + \epsilon \geq \max \left\{ 0, \max_{S'} (v_i(S') - p_i^+(S')) \right\}$$

⁴This is just as well because the values $v_i(S)$ remain private information to agents during in COMBAUCTION, and best-response is the only mode of interaction with an agent.

for any bundle \hat{S}_i allocated to agent i . Then, because $p_{\text{ask}}(S) \geq p_i^+(S) \geq p_{\text{ask}}(S) - \epsilon$, $p(S) = p_{\text{ask}}(S)$, and $x_i(S_i) = 1$ implies that agent i bid for bundle S_i , we have by case analysis that:

$$x_i(S) > 0 \quad \Rightarrow \quad v_i(S) - p(S) + 2\epsilon \geq \max \left\{ 0, \max_{S'} (v_i(S') - p(S')) \right\}$$

Finally, substituting for $p(i)$ from (4.1), we prove ϵ -CS-1:

$$x_i(S) > 0 \quad \Rightarrow \quad p(i) + p(S) \leq v_i(S) + 2\epsilon \quad (\epsilon\text{-CS-1})$$

The second primal CS condition, CS-2, is:

$$y(k) > 0 \Rightarrow \pi - \sum_{S \in k} p(S) = 0 \quad (\text{CS-2})$$

Given (4.2) it states that the allocation must maximize the auctioneer's revenue at prices $p(S)$, over all possible allocations and irrespective of bids received from agents.

We prove that the provisional allocation, computed to maximize revenue based on agents' bids, is sufficient to maintain CS-2 in all rounds.

The following definition is useful:

DEFINITION 4.13 [strict positive price] An ask price $p_{\text{ask}}(S)$ is strictly positive if the price is greater than the ask price for every bundle contained in S , i.e. $p_{\text{ask}}(S) > p_{\text{ask}}(S')$ for all $S' \subset S$.

To see understand how CS-2 can hold in all iterations, notice:

(i) all bundles with strictly-positive prices receive bids in every round.

Agent i with one of the highest losing bid for bundle S in round t will continue to bid for bundle S in rounds $t + 1$. Let $u_i^t(S)$ denote agent i 's utility for bundle S in round t . Then, $u_i^{t+1}(S) = u_i^t(S) - \epsilon$ because the ask price for S increases by ϵ . Also, $u_i^t(S) \geq u_i^t(S')$ for all bundles S' for which agent i did not bid in round t . Hence, with $u_i^t(S') \geq u_i^{t+1}(S')$ because the price of S' can only increase in round $t + 1$, we have $u_i^{t+1}(S) \geq u_i^{t+1}(S') - \epsilon$, and a bid for S' can never exclude a bid for S from agent i 's best-response bids in round $t + 1$. A similar argument can be made for the utility of bundles that did receive a bid from agent i in round t .

(ii) all bundles in revenue-maximizing allocations receive bids from different agents.

No single agent causes the price to increase to its current level on a pair of compatible bundles. This follows because price updates are due to safe bids from agents. It is clear that this cannot happen in a single round. Furthermore, it can be shown by *induction* across rounds that an agent with a myopic best-response bidding strategy cannot increase the price of compatible bundles over a sequence of rounds without submitting unsafe bids in a single round.

Taking (i) with (ii), we have:

$$\sum_{i \in \mathcal{I}} p_i^+(\hat{S}_i) = \max_{k \in K} \sum_{S_i \in k} p_i^+(S_i)$$

Removing personalized prices, because $p_{\text{ask}}(S) \geq p_i^+(S) \geq p_{\text{ask}}(S) - \epsilon$, we have:

$$\sum_{i \in \mathcal{I}} p_{\text{ask}}(\hat{S}_i) + \min\{|\mathcal{G}|, |\mathcal{I}|\}\epsilon \geq \max_{k \in K} \sum_{S_i \in k} p_{\text{ask}}(S_i)$$

otherwise the optimal solution to $\max_{k \in K} \sum_{S_i \in k} p_{\text{ask}}(S_i)$ is a better solution with personalized prices $p_i^+(S)$ than $\hat{S} = (\hat{S}_1, \dots, \hat{S}_I)$.

Finally, substituting for π from (4.2), and because $y(k) > 0$ implies $k = k^*$, and with $p(S) = p_{\text{ask}}(S)$, the partition representing the provisional allocation, we prove ϵ -CS-2:

$$y(k) > 0 \Rightarrow \pi - \sum_{S \in k} p(S) \leq \min\{|\mathcal{G}|, |\mathcal{I}|\}\epsilon \quad (\epsilon\text{-CS-2})$$

The first dual CS condition, (CS-3), is:

$$p(i) > 0 \Rightarrow \sum_{S \subseteq G} x_i(S) = 1 \quad (\text{CS-3})$$

Given (4.1) it states that every agent with positive utility for some bundle at the current prices must receive a bundle in the allocation. (CS-3) is only satisfied during the auction for agents that receive bundles in the provisional allocation, but we prove (CS-3) for all agents when COMBAUCTION terminates.

We immediately have (CS-3) for the agents that satisfy termination condition [T1], because the agents receive a bundle in the provisional allocation. Similarly, we immediately

have (CS-3) for the agents that do not submit a best-response bid, because that implies that they have negative utility for all bundles at the prices. Finally, for agents in termination condition [T2] that receive no bundle but submit the same bids in two successive rounds; these agents must bid at ϵ below the ask price and have values just below ask prices otherwise prices would increase and their bids would change.

Finally, the last pair of dual CS conditions, (CS-4) and (CS-5), are:

$$p(S) > 0 \Rightarrow \sum_{i \in \mathcal{I}} x_i(S) = \sum_{k \in K, S \in k} y(k) \quad (\text{CS-4})$$

$$\pi > 0 \Rightarrow \sum_{k \in K} y(k) = 1 \quad (\text{CS-5})$$

The assignment $y(k^*) = 1$ for the partition $k^* = [\hat{S}_1 \dots \hat{S}_{|\mathcal{I}|}]$ trivially satisfies the right-hand side of both conditions.

Termination. By contradiction, assume the auction never terminates. Informally, [T2] implies that one or more agents must submit different bids in successive rounds, but with myopic best-response bidding this implies that prices must increase and if the auction does not terminate than one or more agents must eventually bid above their values for bundles— a contradiction with myopic best-response.

Putting it all together.

Finally, we prove the worst-case error term in Theorem 4.6 when the auction terminates. Let $S^* = (S_1^*, \dots, S_{|\mathcal{I}|}^*)$ denote the final allocation.

Summing ϵ -CS-1 over all agents in the final allocation, and with $p(i) = 0$ for agents not in the allocation by (CS-3),

$$\sum_{i \in \mathcal{I}} p(i) \leq \sum_{i \in \mathcal{I}} v_i(S_i^*) - \sum_{i \in \mathcal{I}} p(S_i^*) + 2 \min\{|\mathcal{G}|, |\mathcal{I}|\} \epsilon$$

because an allocation can include no more bundles than there are items or agents. Introducing ϵ -CS-2, because $y(k^*) = 1$ for final allocation S_i^* , then $\pi \leq \sum_{i \in \mathcal{I}} p(S_i^*) + \min\{|\mathcal{G}|, |\mathcal{I}|\} \epsilon$.

Adding these two equations, we have:

$$\pi + \sum_{i \in \mathcal{I}} p(i) \leq \sum_{i \in \mathcal{I}} v_i(S_i^*) + 3 \min\{|\mathcal{G}|, |\mathcal{I}|\} \epsilon$$

The left-hand side is the value of the final dual solution, V_{DLP} , and the first-term on the right-hand side is the value of the final primal solution, V_{LP} . We know $V_{\text{LP}}^* \leq V_{\text{DLP}}$, where V_{LP}^* is the value of the optimal primal solution, by the weak-duality property of linear programs.

Thus, because $V_{\text{DLP}} \leq V_{\text{LP}} + 3 \min\{|\mathcal{G}|, |\mathcal{I}|\}\epsilon$, it follows that

$$V_{\text{LP}} \geq V_{\text{LP}}^* - 3 \min\{|\mathcal{G}|, |\mathcal{I}|\}\epsilon$$

Finally, because the primal solution is integral (by construction during COMBAUCTION), it is a feasible and optimal solution to the combinatorial resource allocation problem. ■

4.6.4 Proof: COMBAUCTION(D) and COMBAUCTION(3)

A simple transformation of agents' bids reduces any problem in COMBAUCTION(D) or COMBAUCTION(3) to a safe problem in COMBAUCTION(2). Bertsekas [Bet92] proposes a similar transformation method to derive an AUCTION method for combinatorial optimization problems such as max-flow and the transportation problem, reducing problems to the Assignment problem.

In COMBAUCTION a simple transformation of agents' bids allows COMBAUCTION(3) to be implemented within COMBAUCTION(2) without price-discrimination, and ensures that agents' bids remain safe throughout the auction.

Whenever bids from agent i are not safe in COMBAUCTION(2) we can simulate the price-update rule in COMBAUCTION(3) by introducing a new dummy item that is specific to that agent, call it x_i . This item is concatenated by the auctioneer to all bids from agent i in this round and all future rounds. It has the following effects:

1. The outcome of winner-determination, or the allocative efficiency of the auction, is unchanged because no other agent bids for item x_i .
2. Agent i 's bids are always safe because every bid includes item x_i , and no pair of bids is compatible.
3. The price increases due to bids from agent i are isolated to that agent in all future rounds because all price increases are for bundles that include item x_i .

The optimality of COMBAUCTION(3) follows immediately from the optimality of COMBAUCTION(2) without price discrimination.

4.7 Earlier Primal-Dual Auction Methods

Prior to *i*Bundle there was no method to terminate in competitive equilibrium in the general combinatorial allocation problem (CAP). Table 4.7 summarizes the progress in iterative auction design over the past two decades. Each contribution relaxes assumptions on agent preferences and/or strengthens the equilibrium analysis of the auction. Bidding languages differ in terms of whether agents can bid on items or bundles, whether agents can submit single or multiple bids, and the logic used to combine multiple bids. Prices differ in terms of whether individual items or bundles are priced, and whether prices are anonymous or non-anonymous.

Name	Assumptions	Price structure	Bid structure	Update method	outcome
CK81	linear-additive	items	OR-items	greedy	CE
KC82	GS	items	one bundle	greedy	CE
Ber79–87	unit-demand	items	one item	greedy	CE
DGS86	unit-demand	items	XOR-items	minimal	Vickrey
Aus97	homog subadditive	one price	one bundle	clinch	Vickrey
Aus00	GS	items	one bundle	clinch & unclinch	Vickrey
				$n + 1$ auctions	
GS00	GS	items	XOR-bundle	minimal	min CE
Wur00	monotone	bundles	XOR-bundle	minimal	-
iBundle(2)	safe	bundles	XOR-bundle	greedy	CE
iBundle(d)	monotone	bundles	XOR-bundle	greedy	CE
		non-anonymous			
Adjust	monotone	bundles	XOR-bundle	greedy	min CE
		non-anonymous			
Extend&Adjust	monotone	bundles	XOR-bundle	greedy	(Vickrey)
		non-anonymous			

Table 4.7: Primal-dual auction methods.

All auctions terminate with efficient allocations, but achieve different levels of robustness-to-manipulation. Vickrey payments provide more robustness-to-manipulation than min CE prices, which provide more robustness than CE prices. Termination with Vickrey payments makes myopic best-response a Nash equilibrium of the auction [GS00]. Termination with competitive equilibrium prices provides some incentive-compatibility, at

least in the last round of the auction. Minimal CE prices often coincide with Vickrey payments, but otherwise might be imagined to provide “intermediate” incentive-compatibility properties between that of any CE outcome and the Vickrey outcome.

Computing adjusted prices after termination with *i*Bundle and ADJUST (see Chapter 7), labeled (Adjust) in Table 4.7, implements *minimal* CE prices. These prices support Vickrey payments for both unit-demand and gross-substitutes agent preferences, i.e. capturing all known previous iterative Vickrey auctions for the CAP problem. Moreover, I believe that the extended auction, *i*Bundle Extend&Adjust, is significantly more powerful. I conjecture that the extended auction is an iterative Vickrey auction for *all* CAP instances. The relationship between the final prices in an auction and an auction’s robustness to manipulation is discussed in detail in Chapters 6 and 7.

The main *structural differences* across auctions are in the *bidding-languages*, the *prices*, and the price-update rules. Beyond that, with the exception of Ausubel [Aus97, Aus00], the auctions share the following essential steps:

- (a) announce prices and a provisional allocation
- (b) receive *myopic best-response* bids from agents
- (c) look for a provisional allocation that satisfies every agent
- (d) increase prices on over-demanded items (or bundles) if no solution is found

The auctions differ in the method used to adjust prices across rounds, i.e. the choice of overdemanded set. Price update methods are either *greedy* or *minimal*. In a greedy update rule the price is increased on all over-demanded items (or bundles). In a minimal update rule the price is increased on the smallest set of over-demanded items (or bundles). Prior to the *i*Bundle Extend&Adjust method, the usual approach to compute Vickrey payments was to implement minimal price-updates in each round [DGS86, GS00] of the auction. Minimal price updates are designed to adjust towards *minimal* CE prices, which are equivalent to Vickrey payments in many problems [GS99]. The dual solution that maximizes agent utility and minimizes auctioneer revenue corresponds to the minimal competitive equilibrium solution. In *i*Bundle Extend&Adjust prices are increased beyond minimal CE prices, but adjusted back towards minimal CE prices, and Vickrey payments, after termination.

4.7.1 Assumptions on Agent Preferences

The methods of Bertsekas [Ber79, Ber81, Ber88] and Demange *et al.* [DGS86] assume *unit-demand* preferences, in which each agent demands at most one item:

DEFINITION 4.14 [unit-demand] Agent i 's valuation

$$v_i(S) = \max_{j \in S} v_i(j)$$

The allocation problem for unit-demand preferences is known as the *assignment problem*, with the goal to assign a single item to each agent to maximize value.

Crawford & Knoer [CK81] assume linear-additive agent preferences.

DEFINITION 4.15 [linear-additive] Agent i 's valuation

$$v_i(S) = \sum_{j \in S} v_i(j)$$

Kelso & Crawford [KC82], Gul & Stacchetti [GS00], and Ausubel [Aus00] assume that agents have *gross-substitutes* (GS) preferences. Gross-substitutes states that if the prices were increased from p to p' then the agent would continue to demand any item whose price did not increase (see Definition 4.8). Unit-demand preferences and linear-additive preferences are special cases of GS.

Ausubel's [Aus97] ascending-price auction assumes multiple identical (homogeneous) items and subadditive preferences:

DEFINITION 4.16 [subadditive] Valuation function $v_i(m)$, for $m \geq 0$ units of an item, is subadditive if the marginal value for each additional item is (weakly) decreasing, i.e. if $v_i(n+2) - v_i(n+1) \geq v_i(n+1) - v_i(n)$ for $n \geq 0$ units.

Finally, in *iBundle*, and in *AkBA* [WW00], the only restriction on agent preferences is that they are *monotone*:

DEFINITION 4.17 [monotone] Valuation function $v_i(S)$ is monotone if $v_i(S_1) \geq v_i(S_2)$ for all $S_1 \supseteq S_2$, i.e. free-disposal of additional items.

4.7.2 Relating to Primal-Dual Methods

Bertsekas [Ber79, Ber81, Ber88] was the first to make an explicit connection between primal-dual algorithms and auction mechanisms for combinatorial optimization problems. AUCTION is a primal-dual based method to solve the assignment problem, in which agents have unit-demand preferences. Although AUCTION has a natural auction interpretation, with agents bidding for their favorite item in each round, there is no consideration of agent incentives in Bertsekas' work. His motivation was to develop new algorithms to solve problems in parallel environments, although experimental analysis showed good performance on single processors. See [Ber87] for a text book introduction to the AUCTION algorithm, and [Ber90] for a tutorial.

Bertsekas has also proposed variants of his AUCTION algorithm for other combinatorial optimization problems, including the *transportation problem* [BC89], a variation of assignment with multiple identical items, and the minimal cost flow problem [Ber86]. Bertsekas [Bet92] has recently provided a unified framework of this large body of work, transforming each optimization problem into the assignment problem.

The work of Demange *et al.* [DGS86] is important, as it was the first to consider agent incentives and demonstrate an iterative auction procedure to compute Vickrey payments in the assignment problem. The *DGS* procedure was derived in the context of a linear program formulation for Vickrey payments due to Leonard [Leo83]. Sankaran [San94] appears to provide a primal-dual interpretation of DGS. Recently, Bikchandani *et al.* [BdVSV01] provide an alternative primal-dual interpretation of DGS, with respect to a new linear program formulation of the CAP.

Although primal-dual analysis is not explicit in the work of [CK81, KC82, DGS86, Aus97, Aus00, GS00] a primal-dual explanation can be provided for an appropriate linear programming model [BdVSV01].

Ausubel's auctions [Aus97, Aus00] are quite innovative. In Ausubel [Aus97] the auction maintains one explicit price, but is able to compute multiple prices, one for each agent, that equal the Vickrey payments. Ausubel describes a "clinching" process, whereby the price for items is locked-in during the course of the auction. Bikchandani & Ostroy [BO00] give a primal-dual algorithm and linear program for Ausubel's [Aus97] homogeneous good auction. Bikchandani *et al.* [BdVSV01] propose an alternative derivation based on a network path planning formulation. The clinching rule is reinterpreted as a discount from

a final clearing price in the auction.

Recently Ausubel [Aus00] proposes a complex auction procedure to compute Vickrey payments with GS agent preferences. Ausubel maintains one price for each item, i.e. linear prices, but actually must run $I+1$ auctions (a main auction, and one without each agent in turn) to compute Vickrey payments. The auction's theoretical contribution is significant because it has been observed [GS00] that no single set of linear competitive equilibrium prices can support Vickrey payments with GS agent preferences. Note however that there is a set of non-linear (and perhaps non-anonymous) competitive equilibrium prices that support the Vickrey payments with GS preferences [BdVSV01].

Gul & Stacchetti [GS00] state that *no* dynamic mechanism can reveal sufficient information to implement the Vickrey mechanism with GS preferences. Ausubel's method is outside the spirit of their negative result because of its reliance on multiple auctions. Similarly, my work on *i*Bundle escapes this negative result with non-linear and non-anonymous prices on items. The final price adjust step might also be outside of the spirit of their focus on "monotonic" price adjustment. *i*Bundle with ADJUST will provably compute the efficient allocation and Vickrey payments for GS agent preferences, based on myopic best-response information from agents to an ascending sequence of prices.

Wurman's Ascending k -Bundle Auction (Ak BA) [WW00] family of iterative combinatorial auctions also maintain explicit prices on bundles of items, and were designed with a generalization of the DGS [DGS86] auction in mind. In each round of Ak BA Wurman uses a linear program to increase (and sometimes decrease) prices. A1BA, thought to be the most promising of the family, computes the *maximal* prices that solve a restricted dual problem in each round.