Towards a Universal Test Suite for Combinatorial Auction Algorithms

Kevin Leyton-Brown
Dept. of Computer Science
Stanford University
Stanford, CA 94305
kevinlb@stanford.edu

Mark Pearson
Dept. of Computer Science
Stanford University
Stanford, CA 94305
mpearson@stanford.edu

Yoav Shoham
Dept. of Computer Science
Stanford University
Stanford, CA 94305
shoham@cs.stanford.edu

ABSTRACT

General combinatorial auctions—auctions in which bidders place unrestricted bids for bundles of goods—are the subject of increasing study. Much of this work has focused on algorithms for finding an optimal or approximately optimal set of winning bids. Comparatively little attention has been paid to methodical evaluation and comparison of these algorithms. In particular, there has not been a systematic discussion of appropriate data sets that can serve as universally accepted and well motivated benchmarks. In this paper we present a suite of distribution families for generating realistic, economically motivated combinatorial bids in five broad real-world domains. We hope that this work will yield many comments, criticisms and extensions, bringing the community closer to a universal combinatorial auction test suite.

1. INTRODUCTION

1.1 Combinatorial Auctions

Auctions are a popular way to allocate goods when the amount that bidders are willing to pay is either unknown or unpredictably changeable over time. The rise of electronic commerce has facilitated the use of increasingly complex auction mechanisms, making it possible for auctions to be applied to domains for which the more familiar mechanisms are inadequate. One such example is provided by combinatorial auctions (CA's), multi-object auctions in which bids name bundles of goods. These auctions are attractive because they allow bidders to express *complementarity* and *substitutability* relationships in their valuations for sets of goods. Because CA's allow bids for arbitrary bundles of goods, an agent may offer a different price for some bundle of goods than he offers for the sum of his bids for its disjoint subsets; in the extreme case he may bid for a bundle with the guarantee that he will not receive any of its subsets. An example of complementarity is an auction of used electronic equipment, in which a bidder values a particular TV at x and a particular VCR at y but values the pair at $z > x + y$. An agent with substitutable valuations for two copies of the same book might value either single copy at x, but value the bundle at $z < 2x$. In the special case where $z = x$ (the agent values a second book at 0, having already bought a first) the agent may submit the set of bids $\{bid_1 \text{ XOR } bid_2\}$. By default, we assume that any satisfiable sets of bids that are not explicitly XOR'ed is a candidate for allocation. We call an auction in which all goods are distinguishable from each other a *single-unit CA*. In contrast, in a multi-unit CA some of the goods are indistinguishable (e.g., many identical TVs and VCRs) and bidders request some number of goods from each indistinguishable set. This paper is primarily concerned with single-unit CA's, since most research to date has been focused on this problem. However, when appropriate we will discuss ways that our distributions could be generalized to apply to multi-unit CA's.

1.2 The Computational Combinatorial Auction Problem

In a combinatorial auction, a seller is faced with a set of price offers for various bundles of goods, and his aim is to allocate the goods in a way that maximizes his revenue. (For an overview of this problem, see [8].) This optimization problem is intractable in the general case, even when each good has only a single unit. Because of the intractability of general CA's, much research has focused on subcases of the CA problem that *are* tractable; see [22] and more recently [25]. However, these subcases are very restrictive and therefore are not applicable to many CA domains. Other research attempts to define mechanisms within which general CA's will be tractable (achieved by various trade-offs including bid withdrawal penalties, activity rules and possible inefficiency). Milgrom [15] defines the Simultaneous Ascending Auction mechanism which has been very influential, particularly in the recent FCC spectrum auctions. However, this approach has drawbacks, discussed for example in [6]. In the general case there is no substitute for a completely unrestricted CA. Consequently, many researchers have recently begun to propose algorithms for determining the winners of a general CA, with encouraging results. This wave of research has given rise to a new problem, however. In order to test (and thus to improve) such algorithms, it has been necessary to use some sort of test suite. Since general CA's have never been widely held, there is no data recording the bidding behavior of real bidders upon which such a test suite

may be built. In the absence of such natural data, we are left only with the option of generating artificial data that is representative of the sort of scenarios one is likely to encounter. The goal of this paper is to facilitate the creation of such a test suite.

2. PAST WORK ON TESTING CA ALGORITHMS

2.1 Experiments with Human Subjects

One approach to experimental work on combinatorial auctions uses human subjects. These experiments assign valuation functions to subjects, then have them participate in auctions using various mechanisms [3, 12, 7]. Such tests can be useful for understanding how real people bid under different auction mechanisms; however, they are less suitable for evaluating the mechanisms' computational characteristics. In particular, this sort of test is only as good as the subjects' valuation functions, which in the above papers were hand-crafted. As a result, this technique does not easily permit arbitrary scaling of the problem size, a feature that is important for characterizing an algorithm's performance. In addition, this method relies on relatively naive subjects to behave rationally given their valuation functions, which may be unreasonable when subjects are faced with complex and unfamiliar mechanisms.

2.2 Particular Problems

A parallel line of research has examined particular problems to which CA's seem well suited. For example, researchers have considered auctions for the right to use railroad tracks [5], real estate [19], pollution rights [13], airport time slot allocation [21] and distributed scheduling of machine time [26]. Most of these papers do not suggest holding an unrestricted general CA, presumably because of the computational obstacles. Instead, they tend to discuss alternative mechanisms that are tailored to the particular problem. None of them proposes a method of generating test data, nor does any of them describe how the problem's difficulty scales with the number of bids and goods. However, they still remain useful to researchers interested in general CA's because they give specific descriptions of problem domains to which CA's may be applied.

2.3 Artificial Distributions

Recently, a number of researchers have proposed algorithms for determining the winners of general CA's. In the absence of test suites, some suggested novel bid generation techniques, parameterized by number of bids and goods [24, 10, 4, 8]. (Other researchers have used one or more of these distributions, e.g., [17], while still others have refrained from testing their algorithms altogether, e.g., [16, 14].) Parameterization represents a step forward, making it possible to describe performance with respect to the problem size. However, there are several ways in which each of these bid generation techniques falls short of realism, concerning the selection of which goods and how many goods to request in a bundle, what price to offer for the bundle, and which bids to combine in an XOR'ed set. More fundamentally, however, all of these approaches suffer from failing to model bidders explicitly, and from attempting to represent an economic situation with an non-economic model.

2.3.1 Which goods

First, each of the distributions for generating test data discussed above has the property that all bundles of the same size are equally likely to be requested. This assumption is clearly violated in almost any real-world auction: most of the time, certain goods will be more likely to appear together than others. (Continuing our electronics example, TVs and VCRs will be requested together more often than TVs and printers.)

2.3.2 Number of goods

Likewise, each of the distributions for generating test data determines the number of goods in a bundle completely independently from determining *which* goods appear in the bundle. While this assumption appears more reasonable it will still be violated in many domains, where the expected length of a bundle will be related to which goods it contains. (For example, people buying computers will tend to make long combinatorial bids, requesting monitors, printers, etc., while people buying refrigerators will tend to make short bids.)

2.3.3 Price

Next, there are problems with the pricing[1] schemes used by all four techniques. Pricing is especially crucial: if prices are not chosen carefully then an otherwise hard distribution can become computationally easy.

In Sandholm [24] prices are drawn randomly from either $[0, 1]$ or from $[0, g]$, where g is the number of goods requested. The first method is clearly unreasonable (and computationally trivial) since price is unrelated to the number of goods in a bid—note that a bid for many goods and for a small subset of the same bid will have exactly the same price on expectation. The second is better, but has the disadvantage that average and range are parameterized by the same variable.

In Boutilier et al.[4] prices of bids are distributed normally with mean 16 and standard deviation 3, giving rise to the same problem as the $[0, 1]$ case above.

In Fujishima et al.[10] prices are drawn from $[g(1-d), g(1+d)]$, $d = 0.5$. While this scheme avoids the problems described above, prices are simply additive in g and are unrelated to *which* goods are requested in a bundle, both unrealistic assumptions in some domains.

More fundamentally, Andersson et al.[1] note a critical pricing problem that arises in several of the schemes discussed above. As the number of bids to be generated becomes large, a given short bid will be drawn much more frequently than a given long bid. Since the highest-priced bid for a bundle dominates all other bids for the same bundle, short bids end up being much more competitive. Indeed, it is pointed out that for extremely large numbers of bids a good approximation to the optimal solution is simply to take the best singleton bid for each good. One solution to this problem is to guarantee that a bid will be placed for each bundle at most once (for example, this approach is taken by Sandholm[24]). However, this solution has the drawback that it is unrealistic: different real

---

[1] Most of the existing literature on artificial distributions in combinatorial auctions refers to the monetary amount associated with a bundle as a "price". In Section 3 we will advocate the use of different terminology, but in this section we use the existing term for clarity.

bidders *are* likely to place bids on some of the same bundles.

Another solution to this problem is to make bundle prices superadditive in the number of goods they request—an assumption that may also be reasonable in many CA domains. A similar approach is taken by deVries and Vohra [8], who make the price for a bid a quadratic function of the prices of bids for subsets. For some domains this pricing scheme may result in too large an increase in price as a function of bundle length. The distributions presented in this paper will include a pricing scheme that may be configured to be superadditive or subadditive in bundle length, where appropriate, parameterized to control how rapidly the price offered increases or decreases as a function of bundle length.

2.3.4 XOR bids

Finally, while most of the bid-generation techniques discussed above permit bidders to submit sets of bids XOR'ed together, they have no way of generating meaningful sets of such bids. As a consequence the computational impact of XOR'ed bids has been very difficult to characterize.

3. GENERATING REALISTIC BIDS

While the lack of standardized, realistic test cases does not make it impossible to evaluate or compare algorithms, it does make it difficult to know what magnitude of real-world problems each algorithm is capable of solving, or what features of real-world problems each algorithm is capable of exploiting. This second ambiguity is particularly troubling: it is likely that algorithms would be designed *differently* if they took the features of more realistic[2] bidding into account.

3.1 Prices, price offers and valuations

The term "price" has traditionally been used by researchers constructing artificial distributions to describe the amount offered for a bundle. However, this term really refers to the amount a bidder is made to pay for a bundle, which is of course mechanism-specific and is often not the same as the amount offered. Indeed, it is impossible to model bidders' price offers at all without committing to a particular auction mechanism. In the distributions described in this paper, we will assume a sealed-bid incentive-compatible mechanism, where the price offered for a bundle is equal to the bidder's valuation. Hence, in the rest of this paper, we will use the terms *price offer* and *valuation* interchangeably. Researchers wanting to model bidding behavior in other mechanisms could transform the valuation generated by our distributions according to bidders' equilibrium strategies in the new mechanism.

3.2 The CATS suite

In this paper we present CATS (Combinatorial Auction Test Suite), a suite of distributions for modeling realistic bidding behavior. This suite is grounded in previous research on specific applications of combinatorial auctions, as

described in section 2.1 above. At the same time, all of our distributions are parameterized by number of goods and bids, facilitating the study of algorithm performance. This suite represents a move beyond current work on modeling bidding in combinatorial auctions because we provide an economic motivation for both the contents and the valuation of a bundle, deriving them from basic bidder preferences. In particular, in each of our distributions:

- Certain goods are more likely to appear together than others.

- The number of goods appearing in the bundle is often related to which goods appear in the bundle.

- Valuations are related to which goods appear in the bundle. Where appropriate, valuations can be configured to be subadditive, additive or superadditive in the number of goods requested.

- Sets of XOR'ed bids are constructed in meaningful ways, on a per-bidder basis.

We do not intend for this paper to stand as an isolated statement on bidding in combinatorial auctions, but rather as the beginning of a dialogue. We hope to receive many suggestions and criticisms from members of the CA community, enabling us both to update the distributions proposed here and to include distributions modeling new domains. In particular, our distributions include many parameters, for which we suggest default values. Although these values have evolved somewhat during our development of the test suite, it has not yet been possible to understand the role each parameter plays in the difficulty or realism of the resulting distribution, and our choice may be seen as highly subjective. We hope and expect to receive criticisms about these parameter values; for this reason we include a CATS version number with the defaults to differentiate them from future defaults. The suite also contains a legacy section including all bid generation techniques described above, so that new algorithms may easily be compared to previously-published results. More information on our test suite, including executable versions of our distributions for Solaris, Linux and Windows may be found at *http://robotics.stanford.edu/CATS* .

In section 4, below, we present distributions based on five real-world situations. For most of our distributions, the mechanism for generating bids requires first building a graph representing adjacency relationships between goods. Later, the mechanism uses the graph, generated in an economically-motivated way, to derive complementarity properties between goods and substitutability properties for bids. Of the five real-world situations we model, the first three concern complementarity based on adjacency in (physical or conceptual) space, while the final two concern complementarity based on correlation in time. Our first example (4.1) models shipping, rail and bandwidth auctions. Goods are represented as edges in a nearly planar graph, with agents submitting an XOR'ed set of bids for paths connecting two nodes. Our second example (4.2) models an auction of real estate, or more generally of any goods over which two-dimensional adjacency is the basis of complementarity. Again the relationship between goods is represented by a graph, in this case strictly planar. In (4.3) we relax the planarity assumption from the previous example in order to model arbitrary

---

[2]Previous work characterizes hard cases for weighted set packing—equivalent to the combinatorial auction problem. Real-world bidding is likely to exhibit various regularities, however, as discussed throughout this paper. A data set designed to include the same regularities may be more useful for predicting the performance of an algorithm in a real-world auction.

complementarities between discrete goods such as electronics parts or collectables. Our fourth example (4.4) concerns the matching of time-slots for a fixed number of different goods; this case applies to airline take-off and landing rights auctions. In (4.5) we discuss the generation of bids for a distributed job-shop scheduling domain, and also its application to power generation auctions. Finally, in (4.6), we provide a legacy suite of bid generation techniques, including all those discussed in (2.3) above.

In the description of the distributions that follow, let $rand(a, b)$ represent a real number drawn uniformly from $[a, b]$. Let $rand\_int(a, b)$ represent a random integer drawn uniformly from the same interval. With respect to a given graph, let $e(x, y)$ represent the proposition that an edge exists between nodes x and y. Denote the number of goods in a bundle B as $|B|$. The statement a good g is in a bundle B means that $g \in B$. All of the distributions presented here are parameterized by the number of goods (num_goods) and number of bids (num_bids).

4. CATS IN DETAIL

4.1 Paths in Space

There are many real-world problems involving bidding on paths in space. Generally, this class may be characterized as the problem of purchasing a connection between two points. Examples include truck routes [23], natural gas pipeline networks [20], network bandwidth allocation, and the right to use railway tracks [5].[3] In particular, spatial path problems consist of a set of points and accessibility relations between them. Although the distribution we propose may be configured to model bidding in any of the above domains, we will use the railway domain as our motivating example since it is both intuitive and well-understood.

More formally, we will represent this railroad auction by a graph in which each node represents a location on a plane, and an edge represents a connection between locations. The goods at auction are therefore the edges of the graph, and bids request a set of edges that form a path between two nodes. We assume that no bidder will desire more than one path connecting the same two nodes, although the bidder may value each path differently.

4.1.1 Building the Graph

The first step in modeling bidding behavior for this problem is determining the graph of spatial and connective relationships between cities. One approach would be to use an actual railroad map, which has the advantage that the resulting graph would be unarguably realistic. However, it would be difficult to find a set of real-world maps that could be said to exhibit a similar sort of connectivity and would encompass substantial variation in the number of cities. Since scalability of input data is of great importance

---

[3]Electric power distribution is a frequently discussed real world problem which seems superficially similar to the problems discussed here. However, many of the complementarities in this domain arise from physical laws governing power flow in a network. Consideration of these laws becomes very complex in networks of interesting size. Also, because these laws are taken into account during the construction of power networks, the networks themselves are difficult to model using randomly generated graphs. For these reasons, we do not attempt to model this domain.
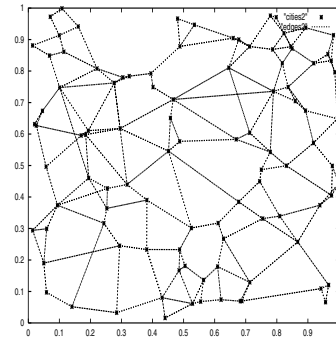

Figure 1: Sample Railroad Graph

to the testing of new CA algorithms, we have chosen to propose generating such graphs randomly. Our technique for generating graphs has various parameters that may be adjusted as necessary; in our opinion it produces realistic graphs with the recommended settings. Figure 1 shows a representative example of a graph generated using our technique.

We begin with num_cities nodes randomly placed on a plane. We add edges to this graph, G, starting by connecting each node to a fixed number of its nearest neighbors. Next, we iteratively consider random pairs of nodes and examine the shortest path connecting them, if any. To compare, we also compute various alternative paths that would require one or more edges to be added to the graph, given a penalty proportional to distance for adding new edges. (We do this by considering a complete graph C, an augmentation of G with new edges weighted to reflect the distance penalty.) If the shortest path involves new edges—despite the penalty—then the new edges (without penalty) are added to G, and replace the existing edges in C. This process models our simplifying assumption that there will exist uniform demand for shipping between any pair of cities, though of course it does not mimic the way new links would actually be added to a rail network. Our technique produces slightly non-planar graphs—graphs on a plane in which edges occasionally cross at points other than nodes. We consider this to be reasonable, as the same phenomenon may be observed in real-world rail lines, highways, network wiring, etc. Determining the "reasonableness" of a graph is of course a subjective task unless more quantitative metrics are used to assess quality; we see the identification and application of such metrics (for this and other distributions) as an important topic for future work.

4.1.2 Generating Bids

Given a map of cities and the connectivity between them, there is the orthogonal problem of modeling bidding itself. We propose a method which generates a set of substitutable bids from a hypothetical agent's point of view. We start with the value to an agent for shipping from one city to another and with a shipping cost which we make equal to the Euclidean distance between the cities. We then place XOR bids on all paths on which the agent would make a profit (i.e., those paths where $utility - cost > 0$). The path's value is random, in (parameterized) proportion to the Euclidean distance between the chosen cities. Since the shipping cost is the Euclidean distance between two cities, we use this as

```
Let num_cities = f(num_goods)
Randomly place nodes (cities) on a unit box
Connect each node to its initial_connections
  nearest neighbors
For i = 1 to num_building_paths:
    C = G
    For every pair of nodes n_1, n_2 ∈ G where
      ¬e(n_1, n_2):
        Add an edge to C of length
          building_penalty ·
          Euclidean_distance(n_1, n_2)
    Choose two nodes at random, and find the
      shortest path between them in C
    If shortest path uses edges that do not
      exist in G:
        For every such pair of nodes
          n_1, n_2 ∈ G add an edge to G with
          length Euclidean_distance(n_1, n_2)
    End If
End For
If total number of edges in G ≠ num_goods,
  restart
```

Figure 2: Graph-Building Technique

```
While num_generated_bids < num_bids:

    Randomly choose two nodes, n_1 and n_2
    d = rand(1, shipping_cost_factor)
    cost = Euclidean_distance(city_1, city_2)
    value = d · Euclidean_distance(city_1, city_2)
    Make XOR bids of value - cost on every path
      from city_1 to city_2 with cost < value
    If there are more than max_bid_set_size such
      paths, bid on the max_bid_set_size paths
      that maximize value - cost.
End While
```

Figure 3: Bid-Generation Technique

the lower bound for value as well, since only bidders with such valuations would actually place bids.

Note that this distribution, and indeed all others presented in this paper, may generate slightly more than num_bids bids. In our experience CA optimization algorithms tend not to be highly sensitive in the number of bids, so we judged it more important to build economically sensible sets of substitutable bids. When generating a precise number of bids *is* important, an appropriate number of bids may be removed after all bids have been generated so that the total will be met exactly.

Note that 1 is used as a lower bound for d because any bidder with $d < 1$ would find no profitable paths and therefore would not bid.

This is CATS 1.0 problem 1. CATS default parameters: $initial\_connections$ = 2, $building\_penalty$ = 1.7, $num\_building\_paths$ = $num\_cities^2/4$, $shipping\_cost\_factor$ = 1.5, $max\_bid\_set\_size$ = 5, and $f(num\_goods) = 0.529689 * NUMGOODS + 3.4329$.

4.1.3 Multi-Unit Extensions: Bandwidth Allocation, Commodity Flow

This model may also be used to generate realistic data for multi-unit CA problems such as network bandwidth allocation and general commodity flow. The graph may be created as above, but with a number of units (capacity) assigned to each edge. Likewise, the bidding technique re-

```
Place nodes at integer vertices (i, j) in a
  plane, where 1 ≤ i, j ≤ ⌈√(num_goods)⌉
For each node n:
    If n is on the edge of the map
        Connect n to as many hv-neighbors as
          possible
    Else
        If rand(0, 1) ≤ three_prob
            Connect n to a random set of
              three of its four hv-neighbors
        Else
            Connect n to all four of its
              hv-neighbors
        While rand(0, 1) ≤ additional_neighbor:

            Connect g to one of its
              d-neighbors, provided that the
              new diagonal edge will not
              cross another diagonal edge
        End While
End For
```

Figure 4: Graph-Building Technique

mains unchanged except for the assignment of a number of units to each bid.

4.2 Proximity in Space

There is a second broad class of real-world problems in which complementarity arises from adjacency in two-dimensional space. An intuitive example is the sale of adjacent pieces of real estate [19]. Another example is drilling rights, where it is much cheaper for an (e.g.) oil company to drill in adjacent lots than in lots that are far from each other. In this section, we first propose a graph-generation mechanism that builds a model of adjacency between goods, and then describe a technique for generating realistic bids on these goods. Note that in this section nodes of the graph represent the goods at auction, while edges represent the adjacency relationship.

4.2.1 Building the Graph

There are a number of ways we could build an adjacency graph. The simplest would be to place all the goods (locations, nodes) in a grid, and connect each to its four neighbors. We propose a slightly more complex method in order to permit a variable number of neighbors per node (equivalent to non-rectangular pieces of real estate). As above we place all goods on a grid, but with some probability we omit a connection between goods that would otherwise represent vertical or horizontal adjacency, and with some probability we introduce a connection representing diagonal adjacency. (We call horizontally- or vertically-adjacent nodes *hv-neighbors* and diagonally-adjacent nodes *d-neighbors*.)

Figure 5 shows a sample real estate graph, generated by the technique described in Figure 4. Nodes of the graph are shown with asterisks, while edges are represented by solid lines. The dashed lines show one set of property boundaries that would be represented by this graph. Note that one node falls inside each piece of property, and that two pieces of property border each other iff their nodes share an edge.

4.2.2 Generating Bids

To model realistic bidding behavior, we generate a set of common values for each good, and private values for each
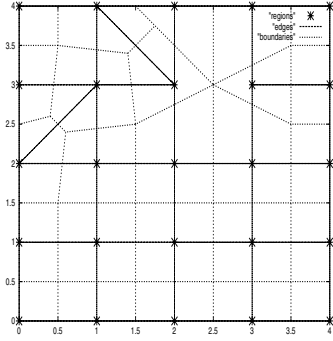
Figure 5: Sample Real Estate Graph

```
Routine Add_Good_to_Bundle(bundle B)
    If rand(0, 1) ≤ jump_prob:
        Add a good g ∉ b to B, chosen
            uniformly at random
    Else:
        Compute s = ∑_{x∉B,y∈B,e(x,y)} pn(x)  [pn()
            is defined below]
        Choose a random node x ∉ B from the
            distribution ∑_{y∈B,e(x,y)} pn(x)/s
        Add x to B
    End If
End Routine
```

Figure 6: Add_Good_to_Bundle for Spatial Proximity

good for each bidder. The common value represents the appraised or expected resale value of each individual good. The private value represents how much one particular bidder values that good, as an offset to the common value (e.g., a private value of 0 for a good represents agreement with the common value). These private valuations describe a bidder's preferences, and so they are used to determine both a value for a given bid and the likelihood that a bidder will request a bundle that includes that good. There are two additional components to each bidder's preferences: a minimum total common value, and a budget. The former reflects the idea that a bidder may only wish to acquire goods of a certain recognized value. The latter reflects the fact that a bidder may not be able to afford every bundle that is of interest to him.

To generate bids, we first add a random good, weighted by a bidder's preferences, to the bidder's bid. Next, we determine whether another good should be added by drawing a value uniformly from [0,1], and adding another good if this value is smaller than a threshold. This is equivalent to drawing the number of goods in a bid from a decay distribution.[4][5] We must now decide which good to add. First we allow a small chance that a new good will be added uniformly at random from the set of goods, without the requirement that it be adjacent to a good in the current bundle B. (This permits bundles requesting unconnected regions of the graph: for example, a hotel company may only wish to build in a city if it can acquire land for two hotels on opposite sides of the city.) Otherwise, we select a good from the set of nodes bordering the goods in B. The probability that some adjacent good n_1 will be added depends on how many edges n_1 shares with the current bundle, and on the bidder's relative private valuations for n_1 and n_2. For example, if nodes n_1 and

---

[4]We use Sandholm's [24] term "decay" here, though the distribution goes by various names—for a description of the distribution please see Section *4.6.1*.

[5]There are two reasons we use a decay distribution here. First, we expect that most bids will request small bundles; a uniform distribution, on the other hand, would be expected to have the same number of bids for bundles of each cardinality. Also, bids for large bundles will often be computationally easier for CA algorithms than bids for small bundles, because choosing the former more highly restricts the future search. Second, we require a distribution where the expected bundle size is unaffected by changes in the total number of goods. Some other distributions, such as uniform and binomial, do not have this property.

n_2 are each connected to B by one edge, and the private valuation for n_1 is twice that for n_2 then the probability of adding n_1 to B, $p(n_1)$, is $2p(n_2)$. Further, if n_1 has 3 edges to nodes in B while n_2 is connected to B by only 1 edge, and the goods have equivalent private values, then $p(n_1) = 3p(n_2)$. Once we have determined all the goods in a bundle we set the price offered for the bundle, which depends on the sum of common and private valuations for the goods in the bundle, and also includes a function that is superadditive (with our parameter settings) in the number of goods.[6] Finally, we generate additional bids that are substitutable for the original bid, with the constraint that each bid in the set requests at least one good from the original bid.

This is CATS 1.0 problem 2. CATS default parameters: *three_prob* = 1.0, *additional_neighbor* = 0.2, *max_good_value* = 100, *max_substitutable_bids* = 5, *additional_location* = 0.9, *jump_prob* = 0.05, *additivity* = 0.2, *deviation* = 0.5, *budget_factor* = 1.5, *resale_factor* = 0.5, and $S(n) = n^{1+additivity}$. Note that *additivity* = 0 gives additive bids, and *additivity* < 0 gives sub-additive bids.

4.2.3 Spectrum Auctions

A related problem is the auction of radio spectrum, in which a government sells the right to use specific segments of spectrum in different geographical areas[18, 2].[7] It is possible to approximate bidding behavior in spectrum auctions by making the assumption that all complementarity arises from spatial proximity.[8] In this case, our spatial proximity model can also be used to generate realistic bidding distributions for spectrum auctions. The main difference between this problem and the real estate problem is that in a spectrum auction each good may have multiple units (frequency bands) for sale. It is insufficient to model this as a multi-unit CA problem, however, if bidders have the constraint that they want the same frequency in each region.[9] Instead,

---

[6]Recall the discussion in Section *2.3.3* motivating the use of superadditive valuations.

[7]Spectrum auctions have not historically been formulated as general CA's, but the possibility of doing so is now being explored.

[8]This assumption would be violated, for example, if some bidders wanted to secure some spectrum in all metropolitan areas. Clearly the problem of realistic test data for spectrum auctions remains an area for future work.

[9]To see why this cannot be modeled as a multi-unit CA, consider an auction for three regions with two units each, and three bidders each wanting one unit of two goods. In

```
For all g, c(g) = rand(1, max_good_value)
While num_generated_bids < num_bids:

    For each good, reset
      p(g) = rand(−deviation ·
      max_good_value, deviation + max_good_value)
```
$pn(g) = \frac{p(g) + deviation \cdot max\_good\_value}{2 \cdot deviation \cdot max\_good\_value}$
```
    Normalize pn(g) so that
```
$\sum_g pn(g) = 1$
```
    B = {}
    Choose a node g at random, weighted by
      pn(), and add it to B
    While rand(0, 1) ≤ additional_location
        Add_Good_to_Bundle(B)
```
$value(B) = \sum_{x \in B}(c(x) + p(x)) + S(|B|)$
```
    If value(B) ≤ 0 on B, restart bundle
      generation for this bidder
    Bid value(B) on B
    budget = budget_factor · value(B)
```
$min\_resale\_value = resale\_factor \cdot \sum_{x \in B} c(x)$
```
    Construct substitutable bids.  For each
      good g_i ∈ B:
        Initialize a new bundle, B_i = {g_i}
        While |B_i| < |B|:
            Add_Good_to_Bundle(B_i)
        Compute
```
$c_i = \sum_{x \in B_i} c(x)$
```
    End For
    Make XOR bids on all B_i where
      0 ≤ value(B) ≤ budget and
      c_i ≥ min_resale_value.
    If there are more than
      max_substitutable_bids such bundles, bid
      on the max_substitutable_bids bundles
      having the largest value
End While
```

Figure 7: Bid-Generation Technique

the problem can be modeled with multiple distinct goods per node in the graph, and bids constructed so that all nodes added to a bundle belong to the same 'frequency'. With this method, it is also easy to incorporate other preferences, such as preferences for different types of goods. For instance, if two different types of frequency bands are being sold, one 5 megahertz wide and one 2.5 megahertz wide, an agent only wanting 5 megahertz bands could make substitutable bids for each such band in the set of regions desired (generating the bids so that the agent will acquire the same frequency in all the regions).

The scheme for generating price offers used in our real estate example may be inappropriate for the spectrum auction domain. Research indicates that while price offers will still tend to be superadditive, this superadditivity may be quadratic in the population of the region rather than exponential in the number of regions [2]. CATS includes a quadratic pricing option that may be used with this problem, in which the common value term above is used as a measure of population. Please see the CATS documentation for more details.

4.3 Arbitrary Relationships

Sometimes complementarities between goods will not be as universal as geographical adjacency, but some kind of reg-

---

the optimal allocation, b_1 gets 1 unit of g_1 and 1 unit of g_2, b_2 gets 1 unit of g_2 and 1 unit of g_3, and b_3 gets 1 unit of g_3 and 1 unit of g_1. In this example there is no way of assigning frequencies to the units so that each bidder gets the same frequency in both regions.

```
Build a fully-connected graph with one node for
  each good
Label each edge from n_1 to n_2 with a weight
  d(n_1, n_2) = rand(0, 1)
```

Figure 8: Graph-Building Technique

```
Routine Add_Good_to_Bundle(bundle B)
    Compute
```
$s = \sum_{x \notin b, y \in B} d(x, y) \cdot pn(x)$
```
    Choose a random node x ∉ B from the
      distribution
```
$\sum_{y \in B} d(x, y) \cdot \frac{pn(x)}{s}$
```
    Add x to B
End Routine
```

Figure 9: Routine Add_Good_to_Bundle for Arbitrary Relationships

ularity in the complementarity relationships between goods will still exist. Consider an auction of different, indivisible goods, e.g. for semiconductor parts or collectables, or for distinct multi-unit goods such as the right to emit some quantity of two different pollutants produced by the same industrial process. In this section we discuss a general way of modeling such arbitrary relationships.

4.3.1 Building the Graph

We express the likelihood that a particular pair of goods will appear together in a bundle as being proportional to the weight of the appropriate edge of a fully-connected graph. That is, the weight of an edge between n_1 and n_2 is proportional to the probability that, having only n_1 in our bundle, we will add n_2. Weights are only proportional to probabilities because we must normalize the sum of all weights from a given good to 1 in order to calculate a probability.

4.3.2 Generating Bids

Our technique for modeling bidding is a generalization of the technique presented in the previous section. We choose a first good and then proceed to add goods one by one, with the probability of each new good being added depending on the current bundle. Note that, since in this section the graph is fully-connected, there is no need for the 'jumping' mechanism described above. The likelihood of adding a new good g to bundle B is proportional to $\sum_{y \in B} d(x, y) \cdot p_i(x)$. The first term $d(x, y)$ represents the likelihood (independent of a particular bidder) that goods x and y will appear in a bundle together; the second, $p_i(x)$, represents bidder i's private valuation of the good x. We implement this new mechanism by changing the routine $Add\_Good\_to\_Bundle()$. We are thus able to use the same techniques for assigning a value to a bundle, as well as for determining other bundles with which it is substitutable.

This is CATS 1.0 problem 3. CATS default parameters: $max\_good\_value = 100$, $additional\_good = 0.9$, $max\_substitutable\_bids = 5$, $additivity = 0.2$, $deviation = 0.5$, $budget\_factor = 1.5$, $resale\_factor = 0.5$, and $S(n) = n^{1+additivity}$.

4.3.3 Multi-Unit Pollution Rights Auctions: Future Work

Bidding in pollution-rights auctions[18, 13] may be modeled through a multi-unit generalization of the technique presented in this section. In such auctions, the government sells companies the right to generate specific amounts of

some pollutant. In the United States, though these auctions are widely used, sulfur-dioxide is the only chemical for which they are the primary method of control. Current US pollution-rights auctions may therefore be modeled as single good multi-unit auctions. If the government were to conduct pollution rights auctions for multiple pollutants in the future, however, bidding would be best-represented as a multi-unit 'Arbitrary Complementarity' problem. The problem belongs to this class because some sets of pollutants are more likely to be produced than others, yet the relationship between pollutants can not be modeled through any notion of adjacency. Should such auctions become viable in the future, we hope that a pollution-rights distribution will be added to CATS .

4.4 Temporal Matching

We now consider real-world domains in which complementarity arises from a temporal relationship between goods. In this section we discuss matching problems, in which corresponding time slices must be secured on multiple resources. The general form of temporal matching includes m sets of resources, in which each bidder wants 1 time slice from each of $j \leq m$ sets subject to certain constraints on how the times may relate to one another (e.g., the time in set 2 must be at least two units later than the time in set 3). Here we concern ourselves with the problem in which $j = 2$, and model the problem of airport take-off and landing rights. Rassenti et al. [21] made the first study of auctions in this domain. The problem has been the topic for much other work; in particular [11] includes detailed experiments and an excellent characterization of bidder behavior.

The airport take-off and landing problem arises because certain high-traffic airports require airlines to purchase the right to take off or land during a given time slice. However, if an airline buys the right for a plane to take off at one airport then it must also purchase the right for the plane to land at its destination an appropriate amount of time later. Thus, complementarity exists between certain pairs of goods, where goods are the right to use the runway at a particular airport at a particular time. Substitutable bids are different departure/arrival packages; therefore bids will only be substitutable within certain limits.

4.4.1 Building the Graph

Departing from our graph-based approach above, we ground this example in the real map of high-traffic US airports for which the Federal Aviation Administration auctions take-off and landing rights, described in [11]. These are the four busiest airports in the United States: La Guardia International, Ronald Reagan Washington National, John F. Kennedy International, and O'Hare International. This map is shown below.

We chose not to use a random graph in this example because the number of bids and goods is dependent on the number of bidders and time slices at the given airports; it is not necessary to modify the number of *airports* in order to vary the problem size. Thus, $num\_cities = 4$ and $num\_times = \lfloor num\_goods/num\_cities \rfloor$.

4.4.2 Generating Bids

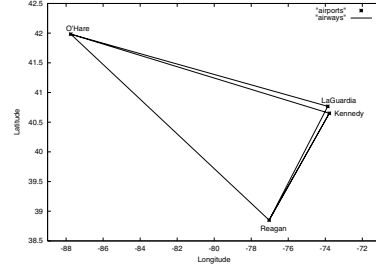Our bidding mechanism presumes that airlines have a certain tolerance for when a plane can take off and land

Figure 10: Map of Airport Locations

($early\_takeoff\_deviation$, $late\_takeoff\_deviation$, $early\_land\_deviation$, $late\_land\_deviation$), as related to their most preferred take-off and landing times ($start\_time$, $start\_time + min\_flight\_length$). We generate bids for all bundles that fit these criteria. The value of a bundle is derived from a particular agent's utility function. We define a utility u_{max} for an agent, which corresponds to the utility the agent receives for flying from $city_1$ to $city_2$ if it receives the ideal takeoff and landing times. This utility depends on a common value for a time slot at the given airport, and deviates by a random amount. Next we construct a utility function which reduces u_{max} according to how late the plane will arrive, and how much the flight time deviates from optimal.

```
Set the average valuation for each city's
  airport:   cost(city) = rand(0, max_airport_value)
Let max_l = length of longest distance between
  any two cities
While num_generated_bids < num_bids:

    Randomly select city₁ and city₂ where
      e(city₁, city₂)
    l = distance(city₁, city₂)
    min_flight_length =
      round(longest_flight_length · 1/max_l)
    start_time =
      rand_int(1, num_times − min_flight_length)
    dev = rand(1 − deviation, 1 + deviation)
    Make substitutable (XOR) bids.  For
      takeoff =
      max(1, start_time − early_takeoff_deviation)
      to min(num_times, start_time +
      late_takeoff_deviation):
        For land = takeoff + min_flight_length
          to
          min(start_time + min_flight_length +
          late_land_deviation, num_times):

            amount_late =
              min(land − (start_time +
              min_flight_length), 0)
            delay =
              land − takeoff − min_flight_length
            Bid dev · (cost(city₁) + cost(city₂)) ·
              delay_coeff^delay ·
              amount_late_coeff^amount_late for
              takeoff at time takeoff at
              city₁ and landing at time land
              at city₂
        End For
    End For
End While
```

Figure 11: Bid-Generation Technique

This is CATS 1.0 problem 4. CATS default parameters: $max\_airport\_value$ = 5, $longest\_flight\_length$ = 10, $deviation$ = 0.5, $early\_takeoff\_deviation$ = 1, $late\_takeoff\_deviation$ = 2, $early\_land\_deviation$ = 1, $late\_land\_deviation$ = 2, $delay\_coeff$ = 0.9, and $amount\_late\_coeff = 0.75$.

4.5 Temporal Scheduling

Wellman et al. [26] proposed distributed job-shop scheduling with one resource as a CA problem. We provide a distribution that mirrors this problem. While there exist many algorithms for solving job-shop scheduling problems, the distributed formulation of this problem places it in an economic context. In the problem formulation from Wellman et al., a factory conducts an auction for time-slices on some resource. Each bidder has a job requiring some amount of machine time, and a deadline by which the job must be completed. Some jobs may have additional, later deadlines which are less desirable to the bidder and so for which the bidder is willing to pay less.

4.5.1 Generating Bids

In the CA formulation of this problem, each good represents a specific time-slice. Two bids are substitutable if they constitute different possible schedules for the same job. We determine the number of deadlines for a given job according to a decay distribution, and then generate a set of substitutable bids satisfying the deadline constraints. Specifically, let the set of deadlines of a particular job be $d_1 < \cdots < d_n$ and the value of a job completed by d_1 be v_1, superadditive in the job length. We define the value of a job completed by deadline d_i as $v_i = v_1 \cdot \frac{d_1}{d_i}$, reflecting the intuition that the decrease in value for a later deadline is proportional to its 'lateness'.

Note that, like Wellman et al., we assume that all jobs are eligible to be started in the first time-slot. Our formulation of the problem differs in only one respect—we consider only allocations in which jobs receive continuous blocks of time. However, this constraint is not restrictive because for any arbitrary allocation of time slots to jobs there exists a new allocation in which each job receives a continuous block of time and no job finishes later than in the original allocation. (This may be achieved by numbering the winning bids in increasing order of scheduled end time, and then allocating continuous time-blocks to jobs in this order. Clearly no job will be rescheduled to finish later than its original scheduled time.) Note also that this problem cannot be translated to a trivial one-good multi-unit CA problem because jobs have different deadlines.

This is CATS 1.0 problem 5. CATS default parameters: $deviation$ = 0.5, $prob\_additional\_deadline$ = 0.9, $additivity = 0.2$, and $max\_length = 10$. Note that we propose a constant maximum job length, because the length of time a job requires should not depend on the amount of time the auctioneer makes available.

4.5.2 Multi-Unit Power Generation Auctions: Future Work

The problem of scheduling power generation is superficially similar to the job-shop scheduling problem described above. In these auctions, electrical power generation companies bid to produce a certain quantity of power for each

```
While num_generated_bids < num_bids:

    l = rand_int(1, max_length)
    d_1 = rand_int(l, num_goods)
    dev = rand(1 - deviation, 1 + deviation)
    cur_max_deadline = 0
    new_d = d_1
    To generate substitutable (XOR) bids.  Do:

        Make bids with price offered
          = dev · l^(1+additivity) · d_1/new_d for all
          blocks [start, end] where start ≥ 1,
          end ≤ new_d, end > cur_max_deadline,
          end - start = l
        cur_max_deadline = new_d
        new_d = rand_int(cur_max_deadline +
          1, num_goods)
    While rand(0, 1) ≤ prob_additional_deadline
End While
```

Figure 12: Bid-Generation Technique

hour of the day. This new problem differs from job-shop scheduling primarily because different kinds of power plants will exhibit very different utility functions, considering different sorts of goods to be complementary. For example, some plants will want to produce for long blocks of time (because they have startup and shutdown costs), others will prefer certain times of day due to labor costs, and still others will have neither restriction [9]. Due to the domain-specific complexity of bidder utilities, the construction of a distribution for this problem remains an area for future work.

4.6 Legacy Distributions

To aid researchers designing new CA algorithms by facilitating comparison with previous work, CATS includes the ability to generate bids according to all previous published test distributions of which we are aware, that are able to scale with the number of goods and bids. Each of these distributions may be seen as an answer to three questions: what number of goods to request in a bundle, which goods to request, and the price offered for a bundle. We begin by describing different techniques for answering each of these three questions, and then show how they have been combined in previously published work.

4.6.1 Number of Goods

Uniform: Uniformly distributed on $[1, num\_goods]$
Normal: Normally distributed with $\mu = \mu\_goods$ and $\sigma = \sigma\_goods$
Constant: Fixed at $constant\_goods$
Decay: Starting with 1, repeatedly increment the size of the bundle until $rand(0, 1)$ exceeds α
Binomial: Request n goods with probability $p^n (1 - p)^{num\_goods - n} \binom{num\_goods}{n}$
Exponential: Request n goods with probability $C \exp^{-n/q}$

4.6.2 Which Goods

Random: Draw n random goods from the set of all goods, without replacement[10]

---

[10] Although in principle the problem of *which* goods to request could be answered in many ways, all legacy distributions of which we are aware use this technique.

4.6.3 Price Offer

Fixed Random: Uniform on $[low\_fixed, hi\_fixed]$.

Linear Random: Uniform on $[low\_linearly \cdot n, hi\_linearly \cdot n]$

Normal: Draw from a normal distribution with $\mu = \mu\_price$ and $\sigma = \sigma\_price$

Quadratic[11]: For each good k and each bidder i set the value $v_k^i = rand(0, 1)$. Then i's price offer for a set of goods S is $\sum_{k \in S} v_k^i + \sum_{k,q} v_k^i v_q^i$.

4.7 Previously Published Distributions

The following is a list of the distributions used in all published tests of which we are aware. In each case we describe first the method used to choose the number of goods, followed by the method used to choose the price offer. In all cases the 'random' technique was used to determine which goods should be requested in a bundle. Each case is labeled with its corresponding CATS legacy suite number; very similar distributions are given similar numbers and identical distributions are given the same number.

[L1] *Sandholm*: Uniform, fixed random with $low\_fixed = 0$, $hi\_fixed = 1$

[L1a] *Andersson et al.*: Uniform, fixed random with $low\_fixed = 0$, $hi\_fixed = 1000$

[L2] *Sandholm*: Uniform, linearly random with $low\_linearly = 0$, $hi\_linearly = 1$

[L2a] *Andersson et al.*: Uniform, linearly random with $low\_linearly = 500$, $hi\_linearly = 1500$

[L3] *Sandholm*: Constant with $constant\_goods = 3$, fixed random with $low\_fixed = 0$, $hi\_fixed = 1$

[L3] *deVries and Vohra*: Constant with $constant\_goods = 3$, fixed random with $low\_fixed = 0$, $hi\_fixed = 1$

[L4] *Sandholm*: Decay with $\alpha = 0.55$, linearly random with $low\_linearly = 0$, $hi\_linearly = 1$

[L4] *deVries and Vohra*: Decay with $\alpha = 0.55$, linearly random with $low\_linearly = 0$, $hi\_linearly = 1$

[L4a] *Andersson et al.*: Decay with $\alpha = 0.55$, linearly random with $low\_linearly = 1$, $hi\_linearly = 1000$

[L5] *Boutilier et al.*: Normal with $\mu\_goods = 4$ and $\sigma\_goods = 1$, normal with $\mu\_price = 16$ and $\sigma\_price = 3$

[L6] *Fujishima et al.*: Exponential with $q = 5$, linearly random with $low\_linearly = 0.5$, $hi\_linearly = 1.5$

[L6a] *Andersson et al.*: Exponential with $q = 5$, linearly random with $low\_linearly = 500$, $hi\_linearly = 1500$

[L7] *Fujishima et al.*: Binomial with $p = 0.2$, linearly random with $low\_linearly = 0.5$, $hi\_linearly = 1.5$

[L7a] *Andersson et al.*: Binomial with $p = 0.2$, linearly random with $low\_linearly = 500$, $hi\_linearly = 1500$

[L8] *deVries and Vohra*: Constant with $constant\_goods = 3$, quadratic

Parkes [17] used many of the test sets described above (particularly those described by Sandholm and Boutilier et al.), but tested with fixed numbers of goods and bids rather than scaling these parameters.

5. CONCLUSION

---

[11]DeVries and Vohra [8] briefly describe a more general version of this price offer scheme, but do not describe how to set all the parameters (e.g., defining which goods are complementary); hence we do not include it here. Quadratic price offers may be particularly applicable to spectrum auctions; see [2].

In this paper we introduced CATS , a test suite for combinatorial auction optimization algorithms. The distributions in CATS represent a step beyond current CA testing techniques because they are economically motivated and model real-world problems. It is our hope that, with the help of others in the CA community, CATS will evolve into a universal test suite that will facilitate the development and evaluation of new CA optimization algorithms.

6. REFERENCES

[1] A. Andersson, M. Tenhnen, , and F. Ygge. Integer programming for combinatorial auction winner determination. In *ICMAS-00*, 2000.

[2] L. Ausubel, P. Crampton, R. McAfee, and J. McMillan. Synergies in wireless telephony: Evidence from the broadband PCS auctions. *Journal of Economics and Management Strategy*, 6(3):497–527, Fall 1997.

[3] J. Banks, J. Ledyard, and D. Porter. Allocating uncertain and unresponsive resources: An experimental approach. *RAND Journal of Economics*, 20:1–23, 1989.

[4] C. Boutilier, M. Goldszmidt, and B. Sabata. Sequential auctions for the allocation of resources with complementarities. Proceedings of IJCAI-99, 1999.

[5] P. Brewer and C. Plott. A binary conflict ascending price (BICAP) mechanism for the decentralized allocation of the right to use railroad tracks. *International Journal of Industrial Organization*, 14:857–886, 1996.

[6] M. Bykowsky, R. Cull, and J. Ledyard. Mutually destructive bidding: The FCC auction design problem. Technical Report Social Science Working Paper 916, California Institute of Technology, Pasadena, 1995.

[7] C. DeMartini, A. Kwasnica, J. Ledyard, and D. Porter. A new and improved design for multi-object iterative auctions. Technical Report Social Science Working Paper 1054, California Institute of Technology, Pasadena, November 1998.

[8] S. DeVries and R. Vohra. Combinatorial auctions: A survey. 2000.

[9] W. Elmaghraby and S. Oren. The efficiency of multi-unit electricity auctions. *IAEE*, 20(4):89–116, 1999.

[10] Y. Fujishima, K. Leyton-Brown, and Y. Shoham. Taming the computational complexity of combinatorial auctions: Optimal and approximate approaches. In *IJCAI-99*, 1999.

[11] D. Grether, R. Isaac, and C. Plott. *The Allocation of Scarce Resources: Experimental Economics and the Problem of Allocating Airport Slots*. Westview Press, Boulder, CO, 1989.

[12] J. Ledyard, D. Porter, and A. Rangel. Experiments testing multiobject allocation mechanisms. *Journal of Economics & Management Strategy*, 6(3):639–675, Fall 1997.

[13] J. Ledyard and K. Szakaly. Designing organizations for trading pollution rights. *Journal of Economic Behavior and Organization*, 25:167–196, 1994.

[14] D. Lehmann, L. O'Callaghan, and Y. Shoham. Truth revalation in rapid, approximately efficient

combinatorial auctions. In *ACM Conference on Electronic Commerce*, 1999.

[15] P. Milgrom. Putting auction theory to work: The simultaneous ascending auction. Technical Report 98-0002, Department of Economics, Stanford University, 1998.

[16] N. Nisan. Bidding and allocation in combinatorial auctions. Working paper, 1999.

[17] D. Parkes. ibundle: An efficient ascending price bundle auction. In *ACM Conference on Electronic Commerce*, 1999.

[18] C. Plott and T. Cason. EPA's new emissions trading mechanism: A laboratory evaluation. *Journal of Environmental Economics and Management*, 30:133–160, 1996.

[19] D. Quan. Real estate auctions: A survey of theory and practice. *Journal of Real Estate Finance and Economics*, 9:23–49, 1994.

[20] S. Rassenti, S. Reynolds, , and V. Smith. Cotenancy and competition in an experimental auction market for natural gas pipeline networks. *Economic Theory*, 4:41–65, 1994.

[21] S. Rassenti, V. Smith, and R. Bulfin. A combinatorial auction mechanism for airport time slot allocation. *Bell Journal of Economics*, 13:402–417, 1982.

[22] M. Rothkopf, A. Pekec, and R. Harstad. Computationally manageable combinatorial auctions. *Management Science*, 44(8):1131–1147, 1998.

[23] T. Sandholm. An implementation of the contract net protocol based on marginal cost calculations. pages 256–262. Proceedings of AAAI-93, 1993.

[24] T. Sandholm. An algorithm for optimal winner determination in combinatorial auctions. In *IJCAI-99*, 1999.

[25] M. Tennenholtz. Some tractable combinatorial auctions. To appear in the proceedings of AAAI-2000, 2000.

[26] M. Wellman, W. Walsh, P. Wurman, and J. MacKie-Mason. Auction protocols for decentralized scheduling. Proceedings of the 18th International Conference on Distributed Computing Systems, 1998.