Addressing Strategic Behavior in a Deployed Microeconomic Resource Allocator

Chaki Ng[†], Philip Buonadonna[*], Brent N. Chun[*], Alex C. Snoeren[‡], Amin Vahdat[‡]
[†]*Harvard* [*]*Intel Research Berkeley* [‡]*UCSD*

ABSTRACT

While market-based systems have long been proposed as solutions for distributed resource allocation, few have been deployed for production use in real computer systems. Towards this end, we present our initial experience using Mirage, a microeconomic resource allocation system based on a repeated combinatorial auction. Mirage allocates time on a heavily-used 148-node wireless sensor network testbed. In particular, we focus on observed strategic user behavior over a four-month period in which 312,148 node hours were allocated across 11 research projects. Based on these results, we present a set of key challenges for market-based resource allocation systems based on repeated combinatorial auctions. Finally, we propose refinements to the system's current auction scheme to mitigate the strategies observed to date and also comment on some initial steps toward building an approximately strategyproof repeated combinatorial auction.

Categories and Subject Descriptors

C.2.4 [**Distributed Systems**]: Distributed Applications

General Terms

Measurement, Design, Economics, Experimentation

Keywords

Strategic Behavior, Resource Allocator, Market-Based Systems

1. INTRODUCTION

Market-based systems have long been proposed as solutions for resource allocation in distributed systems including computational Grids [2, 20], wide-area networking testbeds [9], and peer-to-peer systems [17]. Yet, while the theoretical underpinnings of market-based schemes have made significant strides in recent years, practical integration of market-based mechanisms into real computer systems and empirical observations of such systems under real workloads has remained an elusive goal. Towards this end, we have designed, implemented, and deployed a microeconomic resource allocation system called Mirage [3] for scheduling testbed time on a 148-node wireless sensor network (SensorNet) testbed at Intel Research. The system, which employs a repeated combinatorial auction [5, 14] to schedule allocations, has been in production use for over four months and has scheduled over 312,148 node hours across 11 research projects to date.

In designing and deploying Mirage, we had three primary goals. First, we wanted to validate whether a market-based resource allocation scheme was necessary at all. An economic problem only exists when resources are scarce. Therefore, a key goal was to first measure both resource contention and the range of underlying valuations users place on the resources during periods of resource scarcity. Second, we wanted to observe how users would actually behave in a market-based environment. Much of economic theory is predicated on rational user behavior, which forms the basis for motivating research efforts such as strategyproof mechanism design [19, 4, 6, 15, 16]. With Mirage, we wanted to observe to what extent rationality held and in what ways users would attempt to strategize and game the system. Finally, we wanted to identify what other practical problems would emerge in a deployment of a market based system. In this paper, we report briefly on our first goal while focusing primarily on the second. The third is left for future work.

Empirical results based on four-months of usage have validated the key motivating factors in using an auction-based scheme (i.e., significant resource contention and widely varying valuations) but have also pointed to real world observations of *strategic user behavior*. In deploying Mirage, we made the early decision to base the system on a repeated combinatorial auction known not to be strategyproof. That is, self-interested users could attempt to increase their personal gain, at the expense of others, by not revealing their true value to the system. We made this decision mainly because designing a strategyproof mechanism remains an open, challenging problem and we wanted to deploy a working system and gain experience with real users to address our three goals in a timely manner. Deploying a non-strategyproof mechanism also had the benefit of testing rationality and seeing how and to what extent users would try to game the system. The key contribution of this paper is an analysis of such strategic behavior as observed over a four-month time period and proposed refinements for mitigating such behavior en route to building an approximately strategyproof repeated combinatorial auction.

The rest of this paper is organized as follows. In Section 2,

SIGCOMM'05 Workshops, August 22–26, 2005, Philadelphia, PA, USA.

we present an overview of Mirage including high-level observations on usage over a four-month period. In Section 3, we examine strategic user behavior, focusing on the four primary types of strategies employed by users in the system. Based on these results, Section 4 presents a set of key challenges for market-based resource allocation systems based on repeated combinatorial auctions. As a first step in addressing some of these challenges, we describe refinements to Mirage's current auction scheme that mitigate the strategies observed to date and also comment on some initial steps towards building an approximately strategyproof repeated combinatorial auction for Mirage. Finally, in Section 5, we conclude the paper.

2. THE MIRAGE SYSTEM

SensorNet testbeds are a critical tool for developing and evaluating SensorNet technology in a controlled and instrumented environment. As with many large-scale systems, however, resource management is a key problem given that it is not economical for users to each build and operate their own testbed. In Mirage [3], testbed resources are space-shared and allocated using a repeated combinatorial auction in a closed virtual currency environment. Users compete for testbed resources by submitting bids which specify resource combinations of interest in space/time (e.g., "any 32 MICA2 motes for 8 hours anytime in the next two days") along with a maximum value amount the user is willing to pay. A combinatorial auction is then periodically run to determine the winning bids based on supply and demand while maximizing aggregate utility delivered to users.

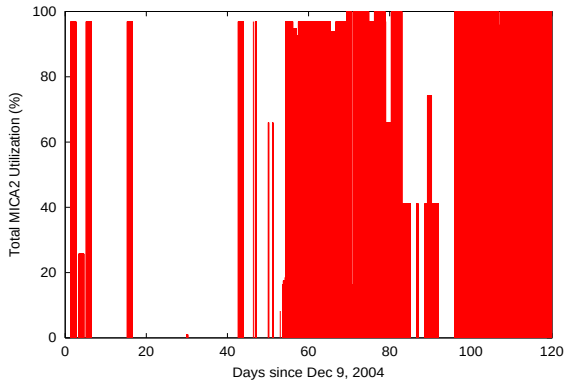

Figure 1: Testbed utilization for 97 MICA2 motes.

In Mirage, resources are allocated using a first-price combinatorial auction which clears every hour. In each round of the auction, a rolling window of future testbed resources is available for allocation with subsets of that window being removed from the pool as resources get allocated. In our initial deployment, we used a 72-hour window and deployed the system on a testbed consisting of 148 nodes (97 MICA2 [1] and 51 MICA2DOT sensor nodes or "motes"). In each round of the auction, users bid for subsets of resources available in the current window. When the system is first brought online, a full 148 node × 72 hour window is available, where each row of the window represents the availability of a particular node across time, and each column represents the availability of the testbed for a given hour. The leftmost column of the window represents node availability for the hour immediately following the auction; these node/hours will never again be available for auction. All other node/hours not allocated at this or previous auctions continue to be offered for sale at subsequent auctions. In each subsequent round (i.e., every hour), portions of the current window get allocated as bids are matched to available resources and a new rightmost 148 node × 1 hour column of resources rolls in and replaces the leftmost 148 node × 1 hour column of resources which expires. There is no time sharing of nodes: given limited local computation and communication power, once a sensor is allocated to a user for a particular time period, it is unavailable to all other users.

In Mirage, users place combinatorial bids specifying resource combinations of interest in space/time along with a maximum value amount the user is willing to pay. More specifically, a bid $b_i = (v_i, s_i, t_i, d_i, f_{min}, f_{max}, n_i, ok_i)$ indicates the user wants any combination of n_i motes from the set ok_i simultaneously for a duration of d_i hours ($d_i \in \{1, 2, \ldots, 32\}$), a start time anywhere between s_i and t_i, and a radio frequency in the range $[f_{min}, f_{max}]$.[1] The user also is willing to pay up to v_i units of virtual currency for these resources. In essence, each bid specifies in a succinct manner what subsets of the resource window would serve as acceptable resources that meet the user's constraints and how important the desired resource allocation is to the user.

We deployed Mirage on December 9, 2004 and the system has been in continuous production use for over four months. In the process, its lifetime has overlapped with several periods of significant resource contention including the SIGCOMM '05 and SenSys '05 conference deadlines. Overall, the system has 18 research projects registered to use the system spanning a variety of academic and commercial institutions. Of these, 11 have actively bid and received time on the system. As of April 8, 2005, the system has received 322 bids, and allocated 312,148 node hours over the testbed's 148 nodes.

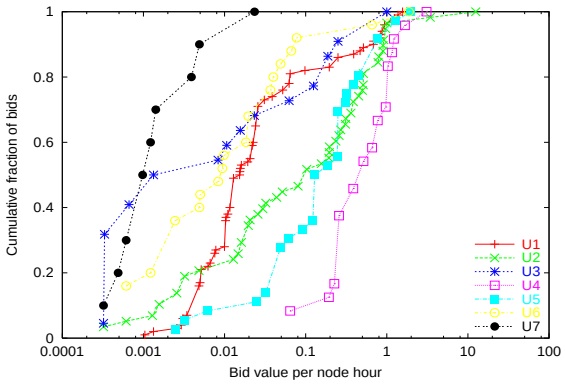

Figure 2: Bid value distributions by user.

As a measure of contention, Figure 1 shows the utilization of the 97 MICA2 motes over the past four months. It depicts periods of significant contention extending over multiple consecutive days, in particular near major deadlines.[2] To quantify user valuations for resources, Figure 2

[1]The frequency constraints are used to schedule testbed allocations such that allocations co-scheduled in time do not collide by using the same radio frequency. In practice, distinct frequencies have not been a scarce resource.

[2]Results for the 51 MICA2DOT motes are similar and omit-

plots distributions of bid values per node hour for the seven most active users in the system. This graph shows that user valuations for testbed resources varied substantially, spanning over four orders of magnitude. Valuations are also distributed relatively evenly across each order of magnitude, suggesting that these ranges are not due to a few anomalous bids but rather to a wide range of underlying user valuations for testbed resources. These dual observations—significant resource contention and a wide range of valuations—support the use of an auction, which is designed precisely to harness such widely varying valuations to compute an efficient and user utility-maximizing node allocation.

Lastly, as another measure of resource contention and the utility of driving resource allocation via user-specified valuations, Figure 3 plots the median per-node clearing price for both MICA2 and MICA2DOT motes over time. To compute these prices, we price an allocated node-hour for a winning bid with value v for n nodes for k hours as v/nk. Unallocated node-hours are assigned a price of 0. For a given hour, we examine all MICA2 motes and plot the median node-hour price for that hour and do the same for MICA2DOT motes. Of particular interest in this graph are the sequence of prices from days 45–60 and days 105–120 (i.e., periods leading up to conference deadlines). These sequences show that the value of testbed resources, as measured by market prices for motes, increases exponentially (logarithmic y-axis) during times of peak contention. This suggests that allowing users to express valuations for resources to drive the resource allocation process is important for making effective use of the testbed (e.g., to distinguish important use from low priority activities). However, it also suggests that users become exponentially desperate to acquire resources as deadlines approach. As it turns out, it is precisely during these times that users will try their hardest to game the system and, therefore, when the efficacy of a market-based mechanism can be best evaluated.

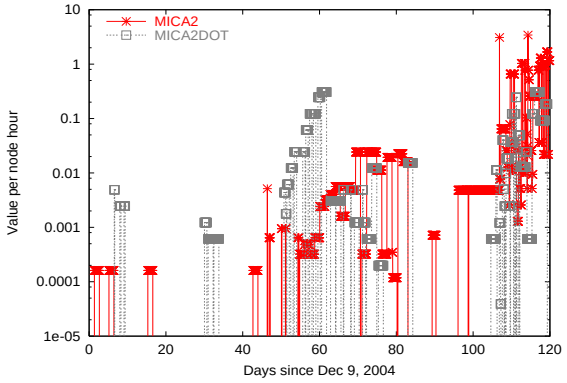

Figure 3: Median node-hour market prices.

3. OBSERVED STRATEGIC BEHAVIOR

During the past four months of operation, Mirage has employed two distinct auction mechanisms and observed four primary types of strategic behavior from users. The first auction mechanism, A1, was deployed from December 9, 2004 to March 28, 2005. During this time period, we observed three different types of strategic behavior (S1-S3), the most recent of which (S3) resulted in significant gaming of the system. In response to the impact of S3, we deployed a second mechanism, A2, on March 29, 2005 (Day 111 in Figures 1 and 3). While A2 mitigated or eliminated the known shortcomings of A1—in particular the vulnerability strategy S3 exploited that prompted the change in the first place—it was soon discovered that A2 remained vulnerable to another strategy, S4, which was predictably discovered and exploited by a motivated user community. We are currently in the process of designing a mechanism to address the weakness in A2 that is abused by S4. Of course, ideally we would develop a provably strategyproof mechanism. However, this remains an open research problem for repeated combinatorial auctions.

In this section, we describe the two auction mechanisms A1 and A2, Mirage's virtual currency policy, the four types of observed strategic behavior S1–S4, and their impact on aggregate utility delivered.

3.1 Auctions and Virtual Currency

Our first auction mechanism, A1 was a first-price, open-bid (i.e., users can see all outstanding bids from competing users) combinatorial auction that cleared every hour based on a greedy algorithm. In each round of auction, the current set of bids was sorted by value per node hour and bids were greedily fit into the remaining portion of the current window of available resources. Like A1, our second auction, A2, was also based on a greedy clearing algorithm. Its key differences were that (i) it was a sealed-bid auction and (ii) it allocated resources over a 148 node × 104 hour window with bid start times constrained to be within the next 72 hours (the reason for this will become apparent when we discuss strategy S3).

In both auctions, winning bids from previous auctions were publicly visible for price feedback and the same virtual currency policy was used. Our virtual currency policy assigns two numbers to each user's bank account: a baseline value and a number of shares. When created, each bank account is initialized to its baseline value. Once funded, a user can then begin to bid and acquire testbed resources through Mirage. In each round of the auction, accounts for winning bids are debited and the proceeds are redistributed through a proportional profit-sharing policy based on bank account share values. The primary purpose of this policy is to reward users who refrain from using the system during times of peak demand and penalize those who use resources aggressively during periods of scarcity. These rewards result in transient bursts of credit and are balanced by another mechanism, a savings tax, to prevent idle users from sitting on large amounts of excess credit forever (a "use it or lose it" policy). In our deployment, an administrator set the virtual currency policy. Bank accounts for external users were assigned baseline and shares value set to 1000, while bank accounts for internal users (U4 and U5) were assigned larger allocations with baseline and share values set to 2000.

3.2 Strategic Behavior

The following are descriptions of the four primary bidding strategies observed over the past four-months.

S1: *underbidding based on current demand.* In A1, all outstanding bids were publicly visible. Consequently, when users would observe a lack of demand, some users would bid correspondingly low amounts rather than their true values. For example, one user would frequently bid 1 or 2 when no

ted for space.

other bids were present. While underbidding in the absence of competition is not a problem *per se*, it does raise two issues. First, if a seller was collecting revenue for profit, such bidding leads to suboptimal outcomes for the seller. Second, should other users enter competing bids before the auction clears, users will need to refine their bids to allow the system to compute an allocation that maximizes aggregate utility. This second problem then leads to strategy S2.

S2: *iterative bidding*. Because users are allowed to modify their bids and A1 was an open auction, iteratively refining one's bid value in response to other users' bid values should, in theory, have no effect on who wins the auction; users with higher valuations—who may also be underbidding—should eventually outbid those with lower valuations after sufficient iteration. The problem is that users do not behave this (rational) way. Usability overhead matters: users in Mirage bid once and perhaps modify their bid a second time. The end result is that inefficiencies may arise since the auction may clear with bid values that are understated. While bidding proxies that automatically adjust user bids in response to other bids are effective in single-good auctions, it is unclear how such proxies could be generally effective in a combinatorial auction without actually implementing the same clearing algorithm used by Mirage (which could be computationally expensive). In summary, S1 and S2 both point toward the need for a *strategyproof* auction mechanism in Mirage. In such an auction, a user's optimal strategy is always to bid truthfully the first time. Thus, rational users will never underbid and iterative bidding is unnecessary.

S3: *rolling window manipulation*. Unlike auctions for tangible goods, resource allocation in computer systems fundamentally involves time, since sharing of resources implies that a resource cannot be assigned to a given user/process forever. In Mirage, we addressed the issue of time by selling resources over a rolling window 72 hours into the future with users able to bid for blocks 1, 2, ..., or 32 hours in length. What we did not anticipate, however, was what would happen when the entire window of resources becomes fully allocated. In this scenario, which was the norm near the recent SenSys '05 deadline, the entire 148 node $\times$ 72 hour window is allocated. A user bidding for, say, 32 hours thus needs to minimally wait 32 hours for 32 new 148 node $\times$ 1 hour columns of resources to become available.

The problem here is that a user can game the system by observing other bids and simply requesting fewer hours. Since 16 columns will roll into the resource window before 32 columns, a user bidding for 16 hours outbids a 32-hour bid independent of each bid's value because resources for the 32-hour bid are not available when the auction clears. Of course, if other users also begin bidding for 16 hours, this opportunity disappears but then moves to durations shorter than 16 hours. In the limit, all users bid for 1-hour blocks, thereby eliminating the possibility of obtaining longer resource allocations which may be critical to the underlying SensorNet experiment. In practice, we observed this type of gaming push winning bid durations down to 2 hours.

With rampant gaming of the system occurring through S3, we responded by implementing and deploying auction A2. As mentioned, a key difference of A2 compared to A1 is that it allocates resources over a 104-hour window with bid start times constrained to be within the next 72 hours. In expanding the window and expanding (while still constraining) the range of start times, A2 eliminates strategy S3. When the entire 148 node $\times$ 72 hour window is allocated, a pending 16-hour bid and a pending 32-hour bid will both have their first opportunity for an allocation when 32 new columns become available. At that point, both the 16-hour bid and the 32-hour bid will have an opportunity to obtain an allocation. Such allocations are then determined by the usual greedy clearing algorithm.

Time	Project	Value	#Nodes	#Hours
04-02-2005 03:58:04	U2	1590	97	32
04-02-2005 05:05:45	U1	5	24	4
04-02-2005 05:28:23	U1	130	40	4
04-02-2005 06:12:12	U1	1	33	4

Table 1: Strategy S4 on 97 MICA2 motes.

S4: *auction sandwich attack*. While A2 eliminated S3 and significantly reduced S1 and S2, it still retained a weakness of A1 that had yet to be discovered and exploited. In the auction sandwich attack, a user exploits two pieces of information: (i) historical information on previous winning bids to estimate the current workload and (ii) the greedy nature of the auction clearing algorithm. In this particular case, a user employs a strategy of splitting a bid for 97 MICA2 motes across several bids, only one of which has a high value per node hour. Since the high value bid is likely to win due to the greedy nature of the auction clearing algorithm and since all other users at the time were all requesting 97 motes (based on the historical information and the fact that the SenSys '05 deadline was imminent requiring experiments at scale), no other bids could backfill the remaining slots; the user's remaining bids would then fit those slots at a low price. An actual occurrence is shown in Table 1. Here, user U1 uses three bids, the main one being a bid with value 130 (value per node hour $130/(4 \cdot 40) = 0.813$) which is used to outbid a bid with value 1590 (value per node hour $1590/(32 \cdot 97) = 0.0512$). Once the high valued 40-node bid has occupied its portion of the resource window, no other 97-node bids can be matched. Consequently, the user backfills on the remaining 57 nodes using two bids: a 24-node bid and a 33-node bid, both at low valuations.

4. CHALLENGES AND REFINEMENTS

Designing an appropriate auction mechanism is key to addressing the above strategies. Specifically, our goals for such a mechanism include: (i) strategyproofness, (ii) computational tractability, and (iii) optimal allocation. The Generalized Vickrey Auction (GVA) [18, 8] is the only known combinatorial mechanism that provides both strategyproofness and optimal allocation. However, it also is computationally intractable as it is NP-hard to calculate the allocations as well as individual payments. Other VCG-based mechanisms exist that replace the allocation algorithms in GVA with approximate ones to provide tractability. In this case, however, strategyproofness is no longer available [16]. These goals are in conflict for VCG and in general[10]. We thus must make certain trade-offs.

With this in mind, we now present a two-phase roadmap for improving Mirage: (i) short-term improvements to the current mechanism that mitigate the effects of existing strategies; and (ii) designing a new mechanism that approximately achieves our three goals simultaneously.

4.1 Ongoing Improvements

Our first improvement is a mixed-integer programming (MIP) formulation as an alternative to the greedy algorithm. This is aimed directly at eliminating strategy S4. While the MIP does not provide strategyproofness, it is able to compute approximately-optimal allocations. Like the GVA, however, the MIP is computationally demanding and thus careful formulation of the MIP and optimizations based on the observed workloads from Mirage will be required to ensure timely clearing of the auction. Our first step is to test and optimize our MIP-based algorithm on auction data from the past four months. We can then run both the MIP alongside the greedy algorithm in parallel and select the higher quality result each time the auction clears.

Second, we can also augment the auction with additional rules and fees to further mitigate strategic behavior. To eliminate S4, two possibilities are to restrict each user to having either one outstanding bid at a time or to mandate that users are not allowed to have multiple overlapping allocations in time. To mitigate S1 and S2, we could add transaction fees. With such fees in place, a user who understates a bid and intends to iteratively refine it will have a disincentive to do so given that each iteration incurs a fee. Finally, another approach to eliminating S4 is to modify the greedy algorithm such that if users do have bids whose allocations could overlap in time, then those potential allocations are considered from lowest to highest value per node hour. In effect, this allows bids for overlapping allocations but creates a disincentive for users to place such bids.

4.2 Towards a Strategyproof Mechanism

Clearly, we need to evaluate our goals and identify where we can make trade-offs in designing a new mechanism. Computational tractability is a fundamental requirement for operational reasons. Strategyproofness or, minimally, making the system hard to manipulate is also key given the behavior we have observed. Finally, our mechanism should compute near-optimal allocations given our compute time budget.

Among the potential mechanisms we can extend, the LOS [12] scheme seems to be a good starting point. It is a fast algorithm as the allocation rule is a greedy mechanism, ranking bids with some "norm" such as value per node hour. The advantage of LOS is its special payment scheme that is tightly linked to the greedy allocation. Essentially, a winner i pays the "norm" of the first bidder denied access times the amount of units (i.e. node hours) that i won. This feature makes it strategyproof. The main downside, however, is that it assumes users are single-minded, meaning that each bidder only cares about a specific set of goods (e.g., a specific list of nodes for specific durations) and they do not value anything else. Unfortunately, this is highly restrictive and contradicts what Mirage currently offers its users, namely the ability to select any subset of nodes for any slots and submit multiple bids. Thus, LOS is vulnerable to S4 and to avoid it we must find a way to extend LOS and its strategyproof property to satisfy complex-bidders.

Realistically, even with a strategyproof LOS scheme for complex bidders there will likely be further strategies we have yet to encounter and that we should consider in our design. For instance, our discussion so far focuses on strategyproofness within a single auction. Across auctions, however, there may be temporal strategies that are possible. For example, in a particular auction, suppose the highest bidder wants all nodes and pays, using GVA payment scheme for simplicity, the next bidder's value. This same bidder may be better off by waiting until the next auction, if the user can still win and face bidders that have even lower values. In this case, the user will gain additional utility due to a lower payment. This, however, may create various problems as total revenue, total value, as well as allocative efficiency across the auctions may be adversely affected.

There are two techniques we can use to address temporal strategies. The first is a "wrapper" scheme such as the one employed by Virtual Worlds (VW) [13] that makes sequences of individually strategyproof auctions (e.g., LOS) strategyproof. What VW does is, after bidder i wins, it tracks what would have happened if i had submitted in a subsequent auction instead. Specifically, it tracks what i *would have paid* in all following auctions during i's patience (i.e., the maximum time i is willing to wait for an allocation) and keeps track of the lowest possible payment. i will instead be charged the lowest payment and will thus have no incentive to temporally game the system. Alternatively, the new class of *online mechanisms*[7, 11] assumes dynamic arrival and departure of bidders and does not hold auctions at fixed intervals. Instead, the mechanism is a continuous scheme that accepts bids as they arrive and makes allocation decisions immediately, thus removing any need to "clear" auctions. The challenge is that the current literature is still restricted to non-combinatorial settings.

5. CONCLUSION

Despite initially using a repeated combinatorial auction known not to be strategyproof, Mirage has shown significant promise as a vehicle for SensorNet testbed allocation. The dual observations of significant resource contention and a wide range of valuations suggest that auction-based schemes can deliver large improvements in aggregate utility when compared to traditional approaches such as proportional share allocation or batch scheduling. Fully realizing these gains, however, requires addressing key problems in strategyproof mechanism design and combinatorial optimization. The temporal nature of computational resources and the combinatorial resource demands of distributed applications adds an additional layer of complexity. Nevertheless, we remain optimistic and believe that a pragmatic mix of theory and practice combined with iterative improvements on real deployments provides one promising avenue toward bringing market-based resource allocation into the mainstream.

6. REFERENCES

[1] Crossbow corporation. `http://www.xbow.com`.

[2] Buyya, R., Abramson, D., and Giddy, J. NimrodG: An Architecture of a Resource Management and Scheduling System in a Global Computational Grid. In *Proceedings of the 4th International Conference on High Performance Computing in Asia-Pacific Region* (May 2000).

[3] Chun, B. N., Buonadonna, P., AuYoung, A., Ng, C., Parkes, D. C., Shneidman, J., Snoeren, A. C., and Vahdat, A. Mirage: A Microeconomic Resource Allocation System for SensorNet Testbeds. In *Proceedings of the 2nd IEEE Workshop on Embedded Networked Sensors* (May 2005).

[4] Clarke, E. H. Multipart pricing of public goods. *Public Choice 2* (1971), 19–33.

[5] de Vries, S., and Vohra, R. V. Combinatorial Auctions: A Survey. *INFORMS Journal on Computing 15* (2003), 284–309.

[6] Groves, T. Incentives in Teams. *Econometrica 41* (1973), 617–631.

[7] Hajiaghayi, M. T., Kleinberg, R., and Parkes, D. C. Adaptive Limited-Supply Online Auctions. In *Proceedings of the 5th ACM Conference on Electronic Commerce* (2004).

[8] Jackson, M. O. Mechanism Theory. In *The Encyclopedia of Life Support Systems*. EOLSS Publishers, 2000.

[9] Lai, K., Huberman, B. A., and Fine, L. Tycoon: A Distributed Market-based Resource Allocation System. Tech. rep., Hewlett Packard, 2004.

[10] Lavi, R., Mu'alem, A., and Nisan, N. Towards a Characterization of Truthful Combinatorial Auctions. In *Proceedings of the 44th Annual Symposium on Foundations of Computer Science* (2003).

[11] Lavi, R., and Nisan, N. Competitive Analysis of Incentive Compatible On-line Auctions. In *Proceedings of the 2nd ACM Conference on Electronic Commerce* (2000), pp. 233–241.

[12] Lehmann, D., O'Callaghan, L. I., and Shoham, Y. Truth Revelation in Approximately Efficient Combinatorial Auctions. *Journal of the ACM 49*, 5 (September 2002), 577–602.

[13] Ng, C., Parkes, D. C., and Seltzer, M. Virtual Worlds: Fast and Strategyproof Auctions for Dynamic Resou rce Allocation. In *Proceedings of the 4th ACM Conference on Electronic Commerce* (2003).

[14] Nisan, N. Bidding and Allocation in Combinatorial Auctions. In *Proceedings of the 2nd ACM Conference on Electronic Commerce* (2000).

[15] Nisan, N., and Ronen, A. Algorithmic Mechanism Design. In *Proceedings of the 31st Annual ACM Symposium on Theory of Computing* (May 1999).

[16] Nisan, N., and Ronen, A. Computationally Feasible VCG Mechanisms. In *Proceedings of the 2nd ACM Conference on Electronic Commerce* (October 2000).

[17] Regev, O., and Nisan, N. The POPCORN Market – an Online Market for Computational Resources. In *Proceedings of the 1st International Conference on Information and Computation Economies* (October 1998).

[18] Varian, H., and MacKie-Mason, J. K. Generalized Vickrey auctions. Tech. rep., University of Michigan, 1995.

[19] Vickrey, W. Counterspeculation, Auctions and Competitive Sealed Tenders. *Journal of Finance* (1961), 8–37.

[20] Wolski, R., Plank, J. S., Brevik, J., and Bryan, T. Analyzing Market-based Resource Allocation Strategies for the Computational Grid. *International Journal of High Performance Computing Applications* (2001), 258–281.