Pricing WiFi at Starbucks – Issues in Online Mechanism Design

Eric J. Friedman[*]
School of Operations Research and Industrial Engineering,
Cornell University, Ithaca, NY 14853.

David C. Parkes[†]
Division of Engineering and Applied Sciences,
Harvard University, Cambridge MA 02138

November 15, 2002

Abstract

We consider the problem of designing mechanisms for online problems in which agents arrive over time and the mechanism is unaware of the agent until the agent announces her arrival. Problems of this sort are becoming extremely common particularly in a wide variety of problems involving wireless networking.

We show how the standard results of mechanism design can be modified to apply to this setting, provide conditions under which efficient and incentive compatible mechanisms exist and analyze several important online models including wireless networks and web serving.

1 Introduction

Mechanism design has a long and successful history in economics and has recently become an active area of research in computer science[NR99, JV01, Pap01, AT01, FS02]. However, most research to date has focussed on static problems, with the notable exceptions related to repeated auctions [VvRM02, BSZ02]. In this paper we will focus on online mechanisms in

[*]This material is based upon work supported by the National Science Foundation under Grant No. ANI–9730162. Email: friedman@orie.cornell.edu. www: http://orie.cornell.edu/ friedman

[†]Email: parkes@eecs.harvard.edu. www: http://www.eecs.harvard.edu/ parkes

which agents arrive over time a problem which is of growing importance due to the increase in wireless services, in which agents are constantly arriving and departing.[1]

For example, consider the pricing of WiFi at a cafe, such as Starbucks.[2] The cafe has a WiFi hub with a limited number of connections. Customers arrive would like to use the WiFi network while enjoying a cup of coffee. However, customers have a wide range of possible valuations for the service, which are known only to themselves. Thus, if our goal is to provide service to those who need it most or to maximize profit, we must have some way to differentiate among users.

This is a classical mechanism design problem with a twist since all computations and analysis must be done online. (We are not allowing customers to call in advance and make reservations.) Thus, in addition to trying to infer agents' valuations, we must also try to infer agents' true arrival time, which may not correspond to their announced arrival time.

In this paper, we will focus on this issue, the announcement of truthful arrival times in order to understand its effect on the general online mechanism design problem. It is an important issue which apparently has not been carefully examined.

One of the obvious effects the arrival issue has on mechanism design is its relation to the revelation principle [MCWG95] which is one of the fundamental ideas in mechanism design. The revelation principle states that in a wide variety of settings, we only need to consider "truthful revelation mechanisms" in which agents truthfully announce their types at time 0; however, this is not even possible in our setting, since most agents do not even exist at time 0!

In this paper we show how to modify the revelation principle for online mechanisms and then analyze the conditions under which efficient mechanisms are possible. In particular we focus on Vickrey-Clarke-Groves (VCG) pricing mechanisms which we use to define strate-

[1]Lavi and Nisan [LN00] have studied an online auction problem with online arrivals; however, they sidestep the timing issues which we focus on because they solve a single, fixed-capacity, allocation problem and simply ensure that prices only ever increase over time. Agents are only able to overstate arrival times, and thus this is sufficient for time strategyproofness. We are interested in dynamic and continuous problems, in which it is unreasonable to assume a pricing function that increases *ad infinitum*!

[2]Starbucks cafe has recently begun offering wireless access in their stores with a variety of pricing plans. (See, http://www.starbucks.com/retail/wireless.asp.)

gyproof (or Bayesian) online mechanisms. One interesting issue is the timing of payments, since they must sometimes be implemented after allocation decisions and thus customers may not know their payment until long after they have departed.

One interesting observation is that while strategyproof online mechanism design is challenging, exactly because decisions must be made online, there is also one aspect in which time is a little easier to deal with from a mechanism design perspective. Agents can claim arrival times *later* than their true arrivals, but cannot claim arrival times before they have actually arrived. Thus, we only need to worry about one-sided manipulation which as we will see simplifies some aspects of the design problem.

Section 2 presents our motivating examples. Section 3 introduces our model, and basic results. In Section 4 we define online VCG mechanisms and analyze the earlier examples, and consider the effect of one-sided manipulation with respect to time. Section 5 introduces Bayesian analysis for online mechanisms constructed around stochastically-optimal online algorithms. In Section 6 we examine the WiFi example in detail, in order to understand both the underlying complexity and the tradeoffs involved in various mechanisms. Section 7 discusses timing issues with respect to payments. Finally, we conclude with some open problems.

2 Motivating Examples

2.1 WiFi at Starbucks

First we consider mechanisms for sharing wireless access among transient users. As seems natural we assume that the mechanism can not know about a customer until they have arrived. For example, we do not allow customers to call in advance.

Customers arrive at the cafe. They sit down at a table and would like wireless access for their entire stay. However, we assume that wireless access is secondary, so the length of their stay is exogenous. We consider two different valuation models. In the first model, the value of customer i for service is a constant, v_i, per unit time. In the second model, the value for customer i for service is a marginal-decreasing valuation, $v_i(\tau)$, for total allocation time, τ.

There is a base station which has m channels on which to serve customers. The base station decides who and when to accept. We consider two different restrictions on our allocation. The first is interruptible agents, in which there are no switching costs and the base station can disconnect and reconnect agents at will. The second is non-interruptible agents, such that once the base station begins serving an agent it cannot end service until the agent departs.

In this model if the current price for service was too high, the customer might delay announcing their presence in the hope that the price might drop due to subsequent departures or arrivals of other customers.

2.2 Web Serving

In this model we consider mechanisms for sharing a web server among users. Assume that page requests arrive at a web server. Request i requires processing time v_i with 100% of the server resource. Assume that all users have the same linear delay cost. In this case, the mechanism must choose the order in which to serve requests.

One again, if the current prices were high, it might be advantageous for a user to delay submitting their request in hopes that prices might fall.

2.3 Ad-hoc Networks

In this example we consider a group of agents who dynamically create their own network. Let X be a metric space and agents arrive with a location $x_i \in X$, and with a value v_i for joining the network and a cost per unit time of maintaining a link with another agent which is equal to c_i times the distance between the agents. (For example, if the cost were due to the power requirements maintaining the link, then the metric we would choose might be the square of the Euclidian distance).

The mechanism constructs a network such that any agent can communicate with any other, i.e., a spanning tree. However, if v_i is too small, then it might be too expensive to connect agent i and the optimal mechanism might delay connecting an agent to the network.

Formally, we are studying a mechanism design version of the prize collecting Steiner tree [GW92, JMP00].

As in the other examples, there might be situations in which it is to an agent's advantage to avoid announcing their arrival.

3 Model and Basic Results

Our notation will be fairly informal, as the gain from precise notation in this setting is outweighed by the complexity necessary to encompass the variety of applicable models. We consider models in which agents have types that include both valuations and arrival time: agent i's type, $\theta_i = (a_i, v_i)$, consists of two elements $a_i \in \Re^+$ and $v_i \in V$ where we interpret a_i as the customer's arrival time and v_i as the time independent elements of the customer's type.[3]

A mechanism, consists of an strategy space S_i which defines the set of strategies available to an agent in the mechanism and an outcome function, $\omega(s)$, which maps strategies $s \in S$, for joint strategy space, S, to outcomes $\omega \in \Omega$ which are detailed descriptions of the system's behavior at any point in time. An agent's strategy is defined in terms of the messages that it sends to the mechanism. We use notation S_{-i} to denote the joint strategy space without agent i.

Clearly, outcome ω must be *time consistent*, which means that the outcome at time t must only depend on messages sent before time t. Similarly, agents can not send messages before they arrive. We will assume that Ω contains only such time consistent, or "online" outcomes. Also, note that we assume that the only remaining information available to the mechanism is distributional information over the types of agents.

An agent's payoff is given by a function $u_i(\omega; a_i, v_i)$ which depends on the outcome and the agent's type. A *social choice function* is a mapping, f, from types to outcomes. We will be interested in online social choice functions, i.e. those for which the outcome at time t is

[3]For simplicity, we assume that departure times are exogenous and either unknown to both the agent and the mechanism, or known to both.

independent of agents that arrive after time t.

A mechanism, together with a set of agents, induces a dynamic game. A strategy for an agent defines a mapping from types to strategy space. We consider two types of implementation. The first is dominant strategy implementation:

Definition 1 (dominant strategy implementation)
1)A strategy, s_i, is a **dominant strategy***, if for all, $s_{-i} \in S_{-i}$, all $s'_i \neq s_i$, and all types θ,*

$$u_i(\omega(s_i(\theta_i), s_{-i}(\theta_{-i})); \theta_i) \geq u_i(\omega(s'_i(\theta_i), s_{-i}(\theta_{-i})); \theta_i)$$

2) A mechanism implements a social choice function, f, in a dominant strategy equilibrium if there is a dominant strategy profile, s^*, such that $\omega(s^*(\theta)) = f(\theta)$ for all θ.

For the second, Bayesian implementation, we assume that there is a probability distribution over types:

Definition 2 (Bayesian implementation)
1) A strategy vector, s^*, is a Bayesian equilibrium, if for all s'_i, and all θ_i,

$$E_{\theta_{-i}}[u_i(\omega(s^*_i(\theta_i), s^*_{-i}(\theta_{-i})); \theta_i)] \geq E_{\theta_{-i}}[u_i(\omega(s'_i(\theta_i), s^*_{-i}(\theta_{-i})); \theta_i)],$$

where the expectation is taken with respect to the distribution over types.
2) A mechanism implements a social choice function, f, in a Bayesian-Nash equilibrium if there is a Bayesian-Nash equilibrium, s^*, for which $\omega(s^*(\theta)) = f(\theta)$, for all θ.

The key result permeating mechanism design is the revelation principle, which implies that any outcome, f, that can be implemented in either dominant-strategy or Bayesian-Nash equilibrium can be *truthfully implemented* by the direct mechanism, in which the agents at time 0 simply state their types (a, v) and the mechanism simply chooses $f(a, v)$ [MCWG95]. In the online setting the analogous result implies that a time dependent online version of the revelation principle applies.

Definition 3 (direct-revelation online mechanisms)
1) A direct-revelation online mechanism is defined as follows: the strategy space for agent i consists of a single announcement of a reported value, $\hat{v}_i \in V$, at a time, $\hat{a}_i$, chosen by the agent (after its arrival). Given an online social choice function f, the mechanism then chooses the outcome that would be chosen by f.
*2) The direct-revelation online mechanism **truthfully implements** social choice function, f, in dominant-strategy (resp. Bayesian equilibrium) if the strategy of truthfully announcing one's type immediately upon arrival is a dominant strategy (resp. Bayesian equilibrium).*

Note that we often refer to a direct revelation mechanism which implements in dominant strategies as being *strategyproof* and one which implements in Bayesian equilibrium as *Bayesian incentive compatible.*

Next we state the online revelation principle, which is the straightforward generalization of the standard revelation principle:

Theorem 1 (Revelation Principle) *For both dominant-strategy and Bayesian equilibrium, if a social choice function can be implemented online, then it can be truthfully implemented by a direct-revelation online mechanism.*

The proof of this result parallels the standard one (e.g. [MCWG95]). In the direct mechanism, one simply considers a trusted referee who, when given the agent's type, chooses the strategy that the agent would have chosen in the original mechanism.

There is an obvious but important corollary to the revelation principle:

Corollary 1 *Only online social choice functions can be implemented by online mechanisms.*

4 VCG Prices

For the remainder of this paper we will specialize to the common setting where agents have *quasilinear* utility functions, $u_i(\omega; \theta_i) = v_i(k; \theta_i) - p_i$, where outcome, ω, is defined in terms of payments, $p = (p_1, p_2, \ldots)$, by each agent and a choice, $k \in \mathcal{K}$, to define the rest of the outcome. We continue to assume that choice set, $\mathcal{K}$, is defined over outcomes that are time-consistent with agent arrivals. For simplicity, we assume a finite set of agents, and define the utilitarian value of k to be the total value, $V(k;\theta) = \sum_i v_i(k;\theta_i)$.

Given some social choice function (SCF), $k(\theta)$, we can define the Vickrey-Clarke-Groves (VCG) prices. Let $\hat{\theta} = (\hat{v}, \hat{a})$ denote the announced types of agents, and $k(\hat{\theta})$ denote the choice implemented with all agents and $k(\hat{\theta}_{-i})$ denote the choice that would be implemented without agent i. Let $V_k(\hat{\theta})$ denote shorthand for $V(k(\hat{\theta}), \hat{\theta})$, and let $V_k(\hat{\theta}_{-i})$ denote shorthand for $V(k(\hat{\theta}_{-i}), \hat{\theta}_{-i})$, the reported value of the choice without i.

Definition 4 (VCG Pricing) *Online VCG pricing given some SCF k implements choice, $k(\hat{\theta})$, given reported types and charges agent i*

$$p_{\text{vcg},i} = v_i(k(\hat{\theta}); \hat{a}_i) - V(k(\hat{\theta}), \hat{\theta}) + g(\hat{\theta}_{-i})$$

where g can be any function. For concreteness (although it does not affect our results) we set $g(\hat{\theta}_{-i}) = V(k(\hat{\theta_{-i}}), \hat{\theta_{-i}})$ as is standard.

Now, in the standard setting, one can easily show that for any SCF, k, the VCG pricing is strategyproof if there is no way for an agent to increase V_k by misstating her type. Thus, if SCF, k, makes choices that maximize $V(k;\theta)$ for all θ, the VCG mechanism will be strategyproof, as discussed in [NR00]. On the contrary, if SCF, k, does not always make a choice that maximizes $V(k;\theta)$, then the mechanism will not be strategy proof if an agent can unilaterally cause a change in the outcome that increases the utilitarian value. If the type and outcome spaces are sufficiently large then one can use these facts to show that SCFs which do not maximize the utilitarian value (over some fixed choice set $\mathcal{K}$) can not be implemented in dominant strategies [MCWG95].

Using these ideas it is straightforward to prove the following.

Theorem 2 *An online VCG mechanism is strategyproof if the online choice rule is offline optimal, in the sense that it maximizes the utilitarian value.*

The proof is analogous to standard proofs for the strategyproofness of VCG mechanisms.

4.1 Offline optimal online algorithms

In each of our motivating examples, assuming that there are no switching costs for either the agents or the servers, the above result allows us to construct optimal strategyproof mechanisms.

For example, in the case of WiFi, with no switching costs and the constant per-unit time valuation model, it is always optimal to serve the m customers with the highest reported valuations in each period. Thus, we can easily construct a VCG mechanism which implements this truthfully. In fact, such a mechanism would be equivalent to a series of $(m+1)$'st price auctions, which set the current price for access.

For WiFi with the marginal-decreasing valuation model, it is always optimal to serve the m customers with the highest reported marginal valuation in each period, given the number of units already allocated to the customers and their reported marginal valuation curves. However, in this case, the online VCG mechanism differs from a series of m'th price auctions.

For web serving, it is well known the serving the request with the shortest remaining processing time (SRPT) is offline optimal [Sch68]. Thus, if the server uses SRPT and the VCG prices, truthtelling will be strategyproof.

Lastly, for the ad-hoc network example, and again with no switching costs, the optimal algorithm simply recomputes the optimal prize collecting Steiner tree in each period, based on the current agent population and based on reported locations, values for connection, and costs to maintain links. The online VCG mechanism can charge according to the current instantaneous VCG prices, as in the constant marginal-value WiFi case. The difficulty for this procedure is that this computation is NP-hard in general [JMP00].

4.2 Time Monotonicity

Although the online nature of our problem makes optimizing more difficult, it also helps, because it limits the ways in which agents might not be truthful. In particular, agents can not state an arrival time which is earlier than their true arrival time. Thus, as we now show, the online algorithm need not be fully optimal, it only needs to be monotonic.

Definition 5 (time monotonicity) *An online choice rule, k, is time monotonic if for any type vector, θ, with arrival time, a_i, for agent i, then $V(k(\theta);\theta) \geq V(k(a_i', v_i', \theta_{-i});\theta)$, for all **delayed** arrivals, $a_i' > a_i$, and all $v_i' \neq v_i$.*

Theorem 3 *The online VCG pricing is strategyproof if and only if the online choice rule is time monotonic.*

The proof is immediate, because if a choice rule does not have this online time-monotonicity property, then an agent can improve its payoff by stalling and reporting arrival time, a_i'. Clearly, off-line optimal choice rules already have this no-stalling property.

4.2.1 Examples

If we consider models with switching costs, then in general there do not exist online algorithms that are offline optimal. However, in some cases we can still construct online time-monotonic algorithms, and therefore strategyproof online mechanisms.

For example, consider the case when arrivals and departures can only occur at discrete intervals. Then when switching costs are small enough compared to the agent values, the algorithm that simply reconfigures the optimal spanning tree, is online time-monotonic for the ad-hoc network problem. An agent can only delaying its entry into the system, and this will have a net negative effect on system value with high values and small switching costs. The loss in total value from such a delay is proportional to the value of the agent, while the gain is at most the reduction in total switching costs. On the other hand, if an agent could announce an earlier arrival then the agent would be able to increase the total (reported) value of the system. However, mechanism is strategyproof because the online algorithm is time-monotonic, and the agent is unable to announce an earlier arrival time.

Even if we consider models with large switching costs, there can still be online time-monotonic algorithms. For example, consider the following algorithm for the ad-hoc problem. Simply connect each new arrival to the closest agent. If the link costs are sufficiently small compared to the agents' values (multiplied by the time increments), then the loss due to delaying for a time increment will always exceed the gain that could arise from a better spanning tree.

5 Bayesian Models

What do we do if there is no online time monotonic algorithm? First, notice that having an online algorithm with the optimal competitive ratio is not good enough because there is a conflict between this worst-case metric and the expected-utility maximizing behavior of agents. Rather, we need stochastically optimal online algorithms, and must also relax strategyproofness to Bayesian incentive-compatibility. It is well known that it is possible to implement many more SCFs under Bayesian implementation than under strategyproofness. For example, any SCF which is Bayesian monotonic can be implemented[PS86, PS89, Jac91]. In this section, we will discuss the Bayesian incentive compatibility of VCG mechanisms in our setting.

Definition 6 (Bayesian optimal) *A SCF is online Bayesian optimal, if at every instant it chooses the outcome that maximizes the expected future utilitarian value.*

Theorem 4 *If a SCF is online Bayesian optimal then the related VCG mechanism is Bayesian-Nash incentive compatible.*

The VCG mechanism is Bayesian-Nash, but not dominant strategy, incentive-compatible, because the Bayesian optimality of the online algorithm depends on correct beliefs about the distribution over agent types, which requires that agents report types truthfully.

Thus, if we can solve the stochastic dynamic program which optimizes the value of the system exactly, then we can use the VCG pricing to convert it into a natural mechanism in which agents reveal their information truthfully in a Bayesian-Nash equilibrium.

As before, we can extend this result to "monotonic SCFs". (Note that this notion of monotonicity is distinct from the idea of Bayesian monotonicity mentioned above.)

Definition 7 (Bayesian time monotonic) *An online choice rule, k, is online Bayesian time monotonic if for any type vector, θ, with arrival time, a_i, for agent i, then $E[V(k(\theta), \theta)] > E[V(k(a_i', v_i', \theta_{-i}), \theta)]$, for all $a_i' > a_i$ and v_i', where the expectation is taken with respect to shared beliefs about the distribution over agent types.*

Theorem 5 *The online VCG mechanism is Bayesian-Nash incentive compatible if and only if the SCF is Bayesian time monotonic.*

6 Mechanisms for WiFi

In this section we examine a specific example in order to understand both the underlying complexity and the tradeoffs involved in various mechanisms.

As a simple mathematical model, assume that customers arrive according to a Poisson process at rate λ. They stay for a time which is exponentially distributed with rate μ. Their value for service is v, which is i.i.d. and uniformly distributed on $[0, 1]$. Assume a base station which can serve at most r customers. In this section we will assume that service is *non-interruptible*. This considerably complicates the mechanism design problem.

We now consider a series of mechanisms in order of increasing complexity. We will see that, even for this simple online problem, the computational difficulties of implementing anything but the simplest mechanism are severe.

6.1 Fixed price

First we consider the simplest reasonable mechanism, the FIFO fixed price mechanism. In this mechanism we set a price p and serve customers in order of their arrival time. Clearly this is strategyproof, since all customers with value greater than p will join the queue as soon as they arrive. (This is also a realistic model of practice.)

To analyze this model, note that this is not a standard $m/m/r$ queuing model since all customers depart at rate μ, not only the ones in service. In fact, the number of customers in the system is given by an $m/m/\infty$ queue, although when we compute the utilitarian value, many of the departures will be of unserved customers, who had to leave the cafe for exogenous reasons.[4]

From the well known formula for an $m/m/\infty$ queue we see that the steady state probability of there being k customers in the system is given by $s_j(\rho') = e^{-\rho'}\rho'^j/j!$ where $\rho' = (1-p)\rho = (1-p)\lambda/\mu$.

Using this a we can compute the steady state social welfare noticing that the average value of v_i for a customer is $(1+p)/2$ and that in state j there are $\min[j,r]$ customers being served. Thus the social welfare is:

$$SW = [(1+p)/2][\sum_{j=1}^{\infty} \min[j,r] s_j(\rho')].$$

When $r = 1$ yields

$$SW = [(1+p)/2](1 - s_0(\rho'))$$

Let p^* be the social welfare maximizing price and note that it only depends on ρ and r. Numerical results for the case with $r = 1$ are shown in Table 1.

6.2 State Dependent Prices

One might try to create another slightly more complex strategyproof mechanism by making the price depend on the number of customers in the system, j, but retain the FIFO servicing. However, we are now constrained in setting prices by strategyproofness in time. Note that with a hybrid approach, in which we assume a probability distribution (arrival process) on types but allow for arbitrary strategies by agents, there are complicated restrictions on prices such that the mechanism will be strategy proof.

First note that if $p_j > p_{j+1}$ then an agent arriving when there are j customers in the system might be tempted to wait for an additional arrival, in order to face a lower price. The critical case here arises when she assumes that other customers will truthfully announce their arrival time. Note that under strategyproofness she can assume that new arrivals will enter the system even if their valuation is less than p_i, something that could be ruled out under

[4]For an introduction to queuing theory see [Wol89].

stronger assumptions, such as iterated dominance [AM92]. Similarly if $p_j > p_{j+1}$ and she arrives when there are $j+1$ customers in the system then she might be tempted to wait for a departure, in which case the critical case arises when all arriving agents decline to enter the system or indefinitely delay announcing their arrival. In order to quantify these constraints one needs to solve a second continuous time Markov chain, which is closely related to the original problem. In addition, the evaluation of the social welfare, given a set of strategyproof prices, is no longer Markovian, since the population of customer values will be dependent on the history of previous states. Thus, even this fairly simple mechanism is extremely difficult to design and evaluate.

If we are interested in truthful Bayesian implementability, then we can use the true arrival process for the computations of the constraints, which simplifies the problem slightly, but still retains the complexities mentioned above.

6.3 On the Optimal Mechanism

Now consider the optimal Bayesian online algorithm. If we do not allow interruptions, then the optimal algorithm is at any time there are less than m jobs in service, to either serve the unserved job with the highest value or wait for a new arrival, by stationarity. Note that this problem is extremely computationally demanding as the state space must include information about every customer in the system. Also, note that it does not help to consider the optimal nonstalling protocol, since this will be suboptimal in the set of stalling protocols and now the VCG mechanism will not be incentive-compatible. Thus, we again see extreme computational difficulties involved in implementing optimal mechanisms.

6.4 Quantifying the Performance of the Simple Mechanism

Given the complexity of analyzing any but the simplest mechanisms, in this section we compute an upper bound on the utilitarian value of the optimal Bayesian mechanism and compare this with the value of the simple fixed price mechanism. We do this by computing the optimal allocation for interruptible agents, which must be larger than that for non-interruptible ones.

The optimal allocation function for interruptible agents is attained by always serving the agents with the largest valuations. For $r = 1$ we compute the steady state social welfare for this function, since by the memoryless properties of the exponential distribution and Poisson process and the fact that valuations are i.i.d. this is given by

$$SW^{opt} = \sum_{j=1}^{\infty} \nu_j s_j$$

where ν_j is the expected value of the maximum of j, v's and $\rho = \lambda/\mu$. Thus, $\nu_j = 1 - 1/(j+1)$ and we see that

$$SW^{opt} = \frac{\rho - 1 + e^{-\rho}}{\rho}.$$

For $r > 1$ this computation is analogous. Numerical results are displayed in Table 1.

ρ	p^*	v_{price}	v_{opt}	loss
0.1	.02	0.0467	0.0483	3%
0.5	0.11	0.199	0.213	7%
1	0.21	0.330	0.368	10%
10	0.71	0.807	0.900	10%

Table 1: Values of different outcomes, $r = 1$.

Note that the losses of using a single price are quite reasonable for all values of ρ. Thus, perhaps the losses due to using a simple suboptimal mechanism like the single price mechanism might be a reasonable tradeoff in order to get both simplicity and strategyproofness, given the complexity of designing and analyzing more complex mechanisms.

7 Timing of Payments

Even ignoring computational issues there are other difficulties with online VCG pricing and online mechanism design in general. This involves the timing of the payments.

Consider the WiFi example when customers have constant value, v_i, per unit time. In this case the online VCG mechanism reduces to a sequence of $(m + 1)$st price auctions (for a server with m channels). However, in general problems, even in problems with optimal online algorithms, the payments cannot be immediately computed.

We can see this effect in the WiFi example in which customers have marginal-decreasing valuations, $v_i(\tau)$, for allocations of time, τ, and also in the SRPT web-serving example. In this case, the VCG payment is a well-behaved function of elapsed time, but in general the effect on system value of agent i can be quite unpredictable for a number of periods ahead. In general, an online VCG mechanism will need to maintain a "virtual world" to represent the state of the system without any particular agent, until the difference between the value of the state of the actual system (with all agents) and the value of the state of the virtual world converges to a constant. The payment to the agent can then be implemented.

In comparison with the strategyproof online VCG mechanisms constructed around offline optimal choice rules, in Bayesian equilibrium implementations with stochastically-optimal online algorithms we can always compute payments immediately, based on the difference between the current estimated long-term value of the system with all agents and the current estimated long-term value of the system without agent i and thus the payment timing issues do not arise for online Bayesian incentive-compatible mechanisms.

8 Conclusions and Open Problems

As we have discussed, one can incorporate online problems into the mechanism design framework. When there exist online choice rules that are offline optimal, then we have both a strategyproof online mechanism and most likely a fast and greedy algorithm. On the other hand, while Bayesian optimal choice rules always *exist*, designing a Bayesian incentive-compatible mechanism seems to require unrealistic amounts of computation. Understanding this issue seems to be of paramount importance.

In situations in which this problem can be solved a secondary difficulty arises – that of computing prices in a timely fashion. For example, it seems clear that in many cases (such as the WiFi example) the effect of a single agent should be short lived and thus an agent's price could be approximately computed in a reasonable amount of time. Another approach would be to try to understand the SCFs which can be implemented with mechanisms that set prices upon an agent's arrival.

Lastly, our analysis emphasizes the importance of understanding mechanism design for computationally intractable problems – apparently a very difficult subject [NR00, AT01]. It is tempting to look for maximal-in-range approximations to online choice problems, either to achieve optimal offline performance with respect to the approximated problem, or to simplify the computation required in Bayesian online algorithms. It is also tempting to ask whether any additional implementation power is achieved with a slight relaxation of the online model, for example allowing the batching of agent arrivals into small groups before making decisions.

References

[AM92] D. Abreu and H. Matsushima. Virtual implementation in iteratively undominated strategies: complete information. *Econometrica*, 60:993–1008, 1992.

[AT01] A. Archer and É. Tardos. Truthful mechanisms for one-parameter agents. In *Proceedings of the 42nd Annual Symposium on Foundations of Computer Science*, 2001.

[BSZ02] A. Blum, T. Sandholm, and M. Zinkevich. Online algorithms for market clearing. In *ACM Symposium on Discrete Algorithms*, pages 971–980, 2002.

[FS02] J. Feigenbaum and S. Shenker. Distributed algorithmic mechanism design: Recent results and future directions. In *Proceedings of the 6th International Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications*, pages 1–13, 2002.

[GW92] Goemans and Williamson. A general approximation technique for constrained forest problems. In *SODA: ACM-SIAM Symposium on Discrete Algorithms (A Conference on Theoretical and Experimental Analysis of Discrete Algorithms)*, 1992.

[Jac91] M. Jackson. Bayesian implementation. *Econometrica*, 59:461–478, 1991.

[JMP00] David S. Johnson, Maria Minkoff, and Steven Phillips. The prize collecting steiner tree problem: theory and practice. In *Symposium on Discrete Algorithms*, pages 760–769, 2000.

[JV01] K. Jain and V. Vazirani. Applications of approximation algorithms to cooperative games. In *Proceedings of the 33rd Annual ACM Symposium on the Theory of Computing*, pages 364–372, 2001.

[LN00] R. Lavi and N. Nisan. Competitive analysis of online auctions. In *Proceedings ACM Conference on Electronic Commerce*, 2000.

[MCWG95] A. Mas-Colell, M. Whinston, and J. Green. *Microeconomic Theory.* Oxford University Press, Oxford, 1995.

[NR99] N. Nisan and A. Ronen. Algorithmic mechanism design. In *Proceedings of the 31st Annual ACM Symposium on the Theory of Computing*, pages 129–140, 1999.

[NR00] Noam Nisan and Amir Ronen. Computationally feasible VCG mechanisms. In *Proc. 2nd ACM Conf. on Electronic Commerce*, pages 242–252, 2000.

[Pap01] C. Papadimitriou. Algorithms, games, and the internet. In *Proceedings of the 33rd Annual ACM Symposium on the Theory of Computing*, page 749753, 2001.

[PS86] A. Postlewaite and D. Schmeidler. Implementation in differential information economies. *Journal of Economic Theory*, 39:14–33, 1986.

[PS89] T. Palfrey and S. Srivastava. Implementation with incomplete information in exchange economies. *Econometrica*, 57:115–134, 1989.

[Sch68] L.E. Schrage. A proof of the optimality of the shortest processing remaining time discipline. *Operations Research*, 16:687–690, 1968.

[VvRM02] G. Vulcano, G. van Ryzin, and C. Maglaras. Optimal dynamic auctions for revenue management. to appear, 2002.

[Wol89] R.W. Wolff. *Stochastic Modelling and the Theory of Queues.* Prentice-Hall international series in industrial and systems engineering. Prentice Hall, Inc., Englewood Cliffs, New Jersey, 1989.