Lectures on Frugal Mechanism Design

Jason Hartline*

Lectures: May 27, 29, 2008
Notes: May 28, 2008; August 21, 2008

These lecture notes cover two lectures from EECS 510, *Algorithmic Mechanism Design*, offered at Northwestern University in the Spring 2008 term. They cover the topic of *frugality* in *mechanism design*. Prerequisites for reading these lecture notes are basic understanding of algorithms and complexity as well as elementary calculus and probability theory. I will also assume that the reader has access to *Lectures on Optimal Mechanism Design*, course notes from the 2005 Fall term of Stanford course CS364B, *Topics in Algorithmic Game Theory*.

Thanks to the students of EECS510. Comments are welcome.

1 Introduction

In these course notes we consider the role combinatorial structure plays on approximation in incentive compatible mechanism design. We will cast these mechanism design questions as ones of *procurement* where the designer is a buyer and they wish to purchase goods or services from agents who are sellers. Recall that for purchasing a single good from any of several agents who can provide the good, the Vickrey auction can be used. The auctioneer solicits bids, purchases the item from the lowest bidder, and pays the second lowest bid value. Of course, the auctioneer's payment is always precisely equal to the true cost of good in absence of the lowest bidder (i.e., the second cheapest good). More generally we might be trying to procure sets of goods that combine in useful ways. What is the extent to which this worst-case property generalizes? When does the VCG mechanism never pay more than the cost of the second cheapest set of goods? When no mechanism can always achieve a total payment of at most the second cheapest set of goods, what is the mechanism that guarantees the best worst-case approximation to it?

Consider the cost a buyer incurs in procuring a set for two paradigmatic set systems: paths and spanning trees. In *path auctions* the agents (representing edges in a graph) in the same path are *complements*, each necessary for the path to be selected. This results in excessive payments as each agent will demand to be reimbursed for the entire path's marginal contribution. In *spanning tree auctions* the agents (again, representing edges in a graph) are *substitutes*, each can be replaced by a single "replacement edge". Each edge can only demand to be paid as much as its replacement edge. This results in very modest payments that are at most the cost of the second cheapest disjoint spanning tree.

Example 1.1 (path auction)

*Northwestern U., Evanston, IL. Email: `hartline@eecs.northwestern.edu`.

```
     1     1     1
s *-----*-----*-----* t
   \_______________/
          10
```

- $G =$ two vertex disjoint s-t paths: $P = (1, 1, 1)$ and $P' = (10)$.
- VCG payments $= 8 \times 3 = 24$.
- second cheapest path $= 10$
- overpayment ratio $= 24/10$.

Example 1.2 (spanning tree auction)

```
     ----
    /    \
   / 1    |
  *----*  | 12
  |   /|  |
1 |  / |1 /
  | /10| /
  |/   |/
  *----*
    11
```

- VCG payments $= 10 + 10 + 11 = 31$.
- second cheapest spanning tree $= 10 + 11 + 12 = 33$.
- overpayment ration $= 31/33$.

We would like to design procurement mechanisms that minimize the total cost paid. In a Bayesian setting if the costs of the agents (a.k.a., edges) are independently distributed from a known distribution, the optimal mechanism is given by Myerson's general construction (See: *Lectures on Optimal Mechanism Design*). Therefore we turn to prior-free settings and attempt to understand mechanisms that are *frugal* in worst-case. As we see from the two examples, above, the VCG mechanism for path auctions sometimes pays more than the second cheapest cost path (edge disjoint), whereas the VCG mechanism for spanning trees (at least in this case) does not. In the sections that follow we explore questions related to this issue:

- Does VCG on spanning trees never cost much more than the second cheapest spanning tree cost?
- How bad can VCG on paths be in comparison to the second cheapest path cost?
- If VCG on paths can be very bad, is there some other mechanism that does well?

2 Path Auctions

As we have seen in the above example, VCG's cost for procuring the cheapest path may actually be more than the cost of the second cheapest (disjoint) path. We start by showing that VCG can be as bad as one might imagine.

Proposition 2.1 *There exists a graph G and edge valuations $\mathbf{v}$ where VCG pays a $\Theta(n)$ factor more than the cost of the second cheapest path.*

Proof: (by construction) Consider the following graph:

```
      0     0           0
s *-----*-----* ... *-----* t
   \____________________/
             1
```

The VCG mechanism selects the top path (which has total cost zero). Each edge in the top path is paid 1. There are $n-1$ such edges resulting in VCG payments totaling $n-1$. The second cheapest path cost is the bottom path with total cost 1. Therefore the ratio of the VCG payments to the second cheapest path cost is $\Theta(n)$. ■

Of course the immediate question to consider from this example is whether this is a flaw of the VCG mechanism or if such a worst-case *overpayment* is an intrinsic property of any incentive compatible mechanism. We show that indeed it is the latter.

Theorem 2.2 *For any incentive compatible mechanism $\mathcal{M}$ and any graph G with two vertex disjoint s-t paths P and P', there is a valuation profile $\mathbf{v}$ such that $\mathcal{M}$ pays an $\Omega(\sqrt{|P||P'|})$ factor more than the cost of the second cheapest path.*

The following corollary is immediate from the theorem – just take a graph with two vertex disjoint s-t paths of length about $n/2$.

Corollary 2.3 *There exists a graph for which any incentive compatible mechanism has a worst-case $\Omega(n)$ factor overpayment.*

Proof of Theorem 2.2: We will prove the theorem for any deterministic incentive compatible mechanism $\mathcal{M}$; though, it can be extended to randomized mechanisms. Let $k = |P|$ and $k' = |P'|$. First we ignore all edges not in P or P' be setting their costs to infinity. Consider edge costs $\mathbf{v}^{(i,j)}$ of the following form (the top path is P and the bottom path is P'):

```
      0     0    v_i    0
    -----*-----*-----*-----
   /                       \
s *                         * t
   \                       /
    -----*-----*-----*-----
      0    v_j    0     0
```

- the cost of the ith edge on P is $v_i = 1/\sqrt{k}$,
- the cost of the jth edge on P' is $v_j = 1/\sqrt{k'}$, and
- all other edges cost zero.

Notice that $\mathcal{M}$ on $\mathbf{v}^{(i,j)}$ must select either all edges in path P or all edges in path P' as winners. We define the directed bipartite graph $G' = (P, P', E')$ on edges in paths P and P'. For any pair of vertices (i, j) in the bipartite graph, there is either a directed edge $(i, j) \in E'$ denoting $\mathcal{M}$ on $\mathbf{v}^{(i,j)}$ selecting path P' (called "forward edges") or a directed edge $(j, i) \in E'$ denoting $\mathcal{M}$ on $\mathbf{v}^{(i,j)}$ selecting path P (called "backwards edges").

Notice that the total number of edges in G' is kk'. With out loss of generality assume that there are more forwards edges than backwards edges, i.e., that there are at least $kk'/2$ forward edges. Since there are k edges in path P, the average number of forward edges per i in P is at least $k'/2$. There must be one edge i with at least this average. Let $N(i)$ with $|N(i)| \geq k'/2$ represent the neighbors of i in the bipartite graph.

Consider the valuation profile $\mathbf{v}^{(i,0)}$ of the following form:

```
      0     0    v_i    0
    -----*-----*-----*-----
   /                       \
s *                         * t
   \                       /
    -----*-----*-----*-----
      0     0     0     0
```

- the cost of the ith edge on P is $v_i = 1/\sqrt{k}$ and
- all other edges cost zero.

Notice that by the definition of $N(i)$, for any $j \in N(i)$, $\mathcal{M}$ on $\mathbf{v}^{(i,j)}$ selects path P'. Since $\mathcal{M}$ is incentive compatible, its allocation rule must be monotone: if agent j is selected when bidding v_j, it must be selected when bidding 0. Therefore, for $j \in N(i)$, since $\mathcal{M}$ selects path P' on input $\mathbf{v}^{(i,j)}$ it must also select path P' on input $\mathbf{v}^{(i,0)}$. Furthermore, note that the payments made by the mechanism to agents j in $N(i)$ are the "maximum bid that j could make and still win" which is at least $v_j = 1/\sqrt{k'}$. Therefore the total payment of $\mathcal{M}$ satisfies:

$$\begin{aligned}\mathcal{M}(\mathbf{v}^{(i,0)}) &= \sum_{j \in P'} \text{maximum winning bid of } j \\ &\geq \sum_{j \in N(i)} 1/\sqrt{k'} \\ &= |N(i)|/\sqrt{k'} \\ &\geq \sqrt{k'}/2.\end{aligned}$$

The last inequality follows from our choice of i to satisfy $|N(i)| \geq k'/2$.

Finally, we use the fact that for $\mathbf{v}^{(i,0)}$ the second cheapest path is P with total cost $1/\sqrt{k}$. The ratio of $\mathcal{M}$'s total payments to the second cheapest path is $\sqrt{kk'}/2$. ■

We can conclude that no mechanism is more frugal than VCG in worst-case and this non-frugality can be as much as a $\Theta(n)$-factor. It is possible to design mechanisms that are better than VCG on non-worst-case inputs; however, the details of this construction are omitted from these notes.

3 Spanning Tree Auctions

We now turn to the problem of procuring a spanning tree in a graph. The set structure of spanning trees are significantly different from paths. Indeed, we will show that the overpayment of VCG for spanning trees is minimal. The following is the main theorm proved in this section.

Theorem 3.1 *The total VCG cost for procuring a spanning tree is at most the cost of the second cheapest disjoint spanning tree.*

To aid our discussion of spanning tree procurement and the analysis of VCG we make the following definitions.

Definition 3.2 The *replacements of* e in a spanning tree T of a graph $G = (V, E)$ are the edges $e' \in E$ that can replace e in the spanning tree T. I.e., $\{e' : T \setminus \{e\} \cup \{e'\}$ is a spanning tree$\}$. The *cheapest replacement of* e is the replacement with minimum cost.

Recall our example from before:

```
      -----
     /     \
    / 1     |
   *----*   | 12
   |   /|   |
 1 |  / |1 /
   | /10| /
   |/   |/
   *----*
     11
```

The MST is given by the three edges with cost 1. The replacements of the left-most 1 in the MST are the edges with cost 10 and 11. The cheapest replacement is therefore the 10 edge.

Definition 3.3 The *bipartite replacement graph* for edge disjoint trees T_1 and T_2 is $G' = (T_1, T_2, E')$ where $(e_1, e_2) \in E'$ if e_2 is a replacement for e_1 in T_1.

Thus, the neighbors $N(e)$ of $e \in T_1$ in the bipartite replacement graph are simply the replacements of e in T_1 (from T_2).

Exercise 3.1 *Construct the bipartite replacement graph for our example above with* $T_1 = \{1, 1, 1\}$ *and* $T_2 = \{10, 11, 12\}$.

With these definitions in hand, the proof of Theorem 3.1 follows from the following steps:

1. The total VCG cost is at most the sum costs of the cheapest replacements of the MST edges.

2. If there is a *perfect matching*[1] in the bipartite replacement graph for cheapest spanning tree T_1 and the second cheapest spanning tree T_2 then the total VCG cost is at most the cost of T_2.

3. There is a perfect matching in the bipartite replacement graph given T_1 and T_2.[2]

3.1 VCG payments and cheapest replacements

Lemma 3.4 *VCG pays each agent (edge) the cost of their cheapest replacement.*

The proof of this lemma is based on the following basic facts about minimum spanning trees.

Fact 3.5 *The cheapest edge across any cut is in the minimum spanning tree.*

Fact 3.6 *The most expensive edge in any cycle is not in any minimum spanning tree.*

Proof of Lemma 3.4: Consider an edge e_1 in the MST T_1. Removal of this edge from T_1 partitions the graph into two sets A and B. The replacements for e_1 are precisely the edges that cross the A-B cut. Since e_1 is the only edge in the MST across the A-B cut, by Fact 3.5 it must be the cheapest edge across the cut. Let e_2 be the second cheapest edge across the A-B cut (and therefore e_1's cheapest replacement).

We claim that if we were to raise the cost of e_1 it would remain in the MST until it exceeds the cost of e_2 after which e_2 would replace it in the MST. This would prove the lemma by implying that the "minimum winning bid" (and thus the VCG payment) of e_1 is precisely the cost of e_2.

First, e_1 is in the MST when bidding less than e_2. This follows from Fact 3.5 as with such a bid, e_1 is the cheapest edge across the A-B cut. Second, e_1 is not in the MST when bidding more than e_2. This follows because there is a cycle in $T_1 \cup \{e_2\}$ that contains both e_1 and e_2. Since e_2 is not in the MST and all other edges in the cycle are, it must be that e_2 is the most expensive edge (by Fact 3.6). However, if e_1's cost is changed to be higher than that of e_2, e_1 would become the most expensive edge in the cycle. Fact 3.6 then implies that with such a cost e_1 could not be in the MST. ■

3.2 Perfect Matchings, the Bipartite Replacement Graph, and VCG payments

Lemma 3.7 *For cheapest and second cheapest spanning trees T_1 and T_2, if there is a perfect matching in the bipartite replacement graph then the VCG payments sum to at most the cost of T_2.*

Proof: Let M be a perfect matching in the bipartite replacement graph for T_1 and T_2. For $e_1 \in T_1$ let $M(e_1)$ denote the edge $e_2 \in T_2$ to which e_1 is matched in M. For $e_1 \in T_1$ let $r(e_1)$ denote the cost of the cheapest replacement for e_1. For ease of notation let $c(e)$ denote the cost (a.k.a., the

[1] **Recall:** a perfect matching in a bipartite graph $G = (A, B, E)$ is a set of edges M such each vertex in A and B has exactly one incident (a.k.a., "matched") edge in M.

[2] To show this we will give a standard proof of Hall's theorem and apply it to MSTs.

value) of edge e. Notice that $r(e_1) \le c(M(e_1))$ as $r(e_1)$ is the cost of the cheapest replacement for e_1 and $c(M(e_1))$ is the cost of replacement $M(e_1)$ which may not be the cheapest.

$$\begin{aligned}
\text{VCG payments} &= \sum_{e_1 \in T_1} r(e_1) && \text{(by Lemma 3.4)} \\
&\le \sum_{e_1 \in T_1} c(M(e_1)) && \text{(since } r(e_1) \le c(M(e_1))\text{)} \\
&= \sum_{e_2 \in T_2} c(e_2). && \text{(since } M \text{ is a matching)}
\end{aligned}$$

Thus, the VCG payments are at least the cost of T_2. ■

3.3 The Bipartite Replacement Graph Has a Perfect Matching

Lemma 3.8 *The bipartite replacement graph for two edge disjoint spanning trees T_1 and T_2 has a perfect matching.*

This proof follows from Hall's Theorem. Hall's Theorem is fundamental to matching theory and a standard proof is given below.

Definition 3.9 The *neighbors* $N(v)$ of a vertex v in a graph $G = (V, E)$ is the set of vertices u that are connected by an edge to v, i.e., $N(v) = \{u : (v, u) \in E\}$. The neighbors of a set of vertices $S \subset V$ is the union of the neighbors of each vertex in the set, i.e., $N(S) = \bigcup_{v \in S} N(v)$.

Definition 3.10 (Hall's condition) A bipartite graph $G = (A, B, E)$ satisfies *Hall's condition* if all subsets $S \subseteq A$ satisfy $|S| \le |N(S)|$.

Theorem 3.11 (Hall's Theorem) *For bipartite graph $G = (A, B, E)$, G has a perfect matching if and only if it satisfies Hall's condition (i.e., all subsets $S \subseteq A$ satisfy $|S| \le |N(S)|$).*

Proof: The "only if" direction is trivial. Suppose there is a perfect matching M in G. For any $S \subset A$, each $v \in S$ is matched to a distinct vertex $u \in B$. $N(S)$ contains at least these vertices and there are $|S|$ of them. Clearly then, $|S| \le |N(S)|$.

We now argue the "if" direction by induction on $|A|$. The premise is that Hall's condition holds and we wish to show that this implies that a perfect matching exists. The base case is trivial. if $|A|$ is one or zero then Hall's condition trivially implies a perfect matching.

We now make the inductive hypothesis that all bipartite graphs $G' = (A', B', E')$ with $|A'| = |B'| < k$ that satisfy Hall's condition contain perfect matchings. From this we show that the same holds for any graph $G = (A, B, E)$ with $|A| = |B| = k$.

Case 1: When a "strong Hall's condition" holds: for all $S \subseteq A$, $|N(S)| \ge |S| + 1$.

Pick any $v \in A$ and $u \in N(v)$. Match v to u. Consider the residual graph G' of G with u and v removed. Our "strong Hall's condition" on G implies that Hall's condition holds on G': in going from G to G' the neighborhood size of any set $S \subset A$ in G' is at most one less then that of S in G. Our inductive hypothesis applied to G' implies a perfect matching exists. Adding the matched edge (v, u) to the perfect matching of G' gives a perfect matching of G.

Case 2: When "strong Hall's condition" does not hold: exists $S \subset A$, $|N(S)| = |S|$.

Consider two induced graphs: $G' = (S, N(S), E)$ and $G'' = (A \setminus S, B \setminus N(S), E)$.

1. Hall's condition holds for G'.

   This follows since all edges incident on S are included in the induced graph.

2. Hall's condition holds for G''.

   This is not immediately obvious as vertices in $A'' = A \setminus S$ may have neighbors in $N(S)$, edges to which have been removed in G''. Consider some subset $S'' \subset A''$ and let $N''(\cdot)$ represent the neighbors with respect to G'' (whereas $N(\cdot)$ represents the neighbors with respect to the original graph G). Notice that the neighbors of S'' in G'' are simply the neighbors of $S'' \cup S$ in G not including the neighbors of S. Formally:

   $$N''(S'') = N(S'' \cup S) \setminus N(S)$$

   Thus,

   $$\left|N''(S)\right| = \left|N(S'' \cup S)\right| - |N(S)|$$

   But, by Hall's condition on G, $|N(S'' \cup S)| \geq |S'' \cup S|$; and by our Case 2 assumption, $|N(S)| = |S|$. So,

   $$\begin{aligned} \left|N''(S)\right| &\geq \left|S'' \cup S\right| - |S| \\ &= \left|S''\right|. \end{aligned}$$

   Therefore, Hall's condition holds on G''.

3. G has a perfect matching.

   Our inductive hypothesis and Hall's condition holding on both G' and G'' implies that there are perfect matchings M' and M'' in G' and G'' respectively. The union of these matchings $M = M' \cup M''$ is a perfect matching of G.

That both cases imply a perfect matching in G completes the proof. ■

Now we are ready to argue that Hall's condition holds in the bipartite replacement graph for any T_1 and T_2; therefore, it contains a perfect matching.

Proof of Lemma 3.8: Consider some subset $S_1 \subset T_1$. Let $k = |S_1|$. When we remove S_1 from T_1 the remaining tree edges do not span G. In particular there are exactly $k+1$ connected components. We can view these connected components as a "super-nodes" and S_1 as a spanning tree of these super-nodes. Let $S_2 \subset T_2$ be the set of edges from T_2 that connect any pair of super-nodes. We now make two arguments.

1. Any $e_2 \in S_2$ is a replacement for some $e_1 \in S_1$, i.e., $S_2 \subseteq N(S_1)$.[3]

   Consider any $e_2 \in S_2$. By definition, e_2 connects two super-nodes. S_1 is a spanning tree of these super-nodes which implies that there is exactly one path in S_1 that connects them. The edge e_2 is a replacement for any edge e_1 in this path.

[3] Actually, $N(S_1) = S_2$, but we only need one direction.

2. $|S_2| \geq k$.

   Since T_2 spans the original graph and S_2 is precisely the set of edges from T_2 that are between super-nodes, S_2 must span the graph of super-nodes. There are $k+1$ super-nodes therefore such a set of spanning edges must be of size at least k. We conclude that $|S_2| \geq k$.

Combining the above two arguments: $|N(S_1)| \geq |S_2| \geq k = |S_1|$. Thus, Hall's condition holds for the bipartite replacement graph. Hall's Theorem then implies a perfect matching exists. ■

3.4 Summary

The proof of Theorem 3.1 follows from Lemmas 3.4, 3.7, and 3.8.

3.5 Generalizations

The fact that VCG has no overpayment for spanning tree auctions raises an interesting question. For what kinds of set systems does VCG never have an overpayment? Spanning trees, it turns out, are a special case of a large class of set systems known as *matroids*. Generally, matroids are set systems where analogs of Fact 3.5 and Fact 3.6 hold. These facts imply a *single-replacement* property. From these, all of our results for spanning trees can be generalized to matroid set systems.

One of the most relevant set systems for auction design is the *transversal matroid*. Transversal matroids correspond to *assignment problems* (a.k.a., matching problems). Imagine selling houses to people when each person has a set of houses they desire. Then the task is to allocate the houses such that each person gets one of their desired houses. Structurally, this problem is a bipartite matching problem. When we consider the sets of people who can simultaneously be assigned to houses, these sets are a matroid. In a procurement settings, we could imagine trying to hire a sports team, e.g., for baseball. Each position must be filled, some players can play several positions, but not at once; the sets of people that can simultaneously be playing on the same team form a transversal matroid.

Generalizing the arguments for spanning trees to matroids is only half of the answer to the question of VCG overpayment. We want to know all of the set systems for which VCG overpayment is minimal. It turns out there is a very precise answer to this, but stating it requires moving beyond the framework discussed in these notes. Instead we summarize.

Proposition 3.12 *There is a very precise sense in which matroid set systems are the only set systems for which VCG has no overpayment.*

4 Guide to the Literature

Archer and Tardos [1] initiated the study of frugality by showing that no mechanism from a large class of incentive compatible mechanisms can approximate the second cheapest disjoint path to better than a linear factor. Elkind et al. [3] generalized this impossibility result to include all incentive compatible mechanisms. Talwar [5] extended the analysis of spanning tree frugality to matroid set systems and also considered the frugality of a number of other interesting set systems including vertex cover, edge cover, facility location, vertex cut, and bipartite matching. Karlin et al. [4] gave a clean analysis framework for the consideration of frugality and showed that matroids are precisely the set systems for which VCG is frugal (Proposition 3.12. Further, they showed that for path procurement, where the lower bound on overpayment depends on the graph structure, it is possible to design a mechanism to match, up to a constant factor, the lower bound of Theorem 2.2. Finally, the reader may have noticed that the discussion of prior-free cost minimization in procurement has been technically unrelated to the conceptually related task of prior-free profit maximization (See: *Lectures on Optimal Mechanism Design*). Cary et al. [2] unify these two research areas by giving random sampling based auctions for procuring multiple sets from a matroid set system, and by showing that it is unlikely for a similar result to be possible for path auctions.

References

[1] A. Archer and E. Tardos. Frugal path mechanisms. In *Proc. 13th ACM Symp. on Discrete Algorithms*, pages 991–999. ACM/SIAM, 2002.

[2] M. Cary, A. Flaxman, J. Hartline, and A. Karlin. Auctions for structured procurement. In *Proc. 19th ACM Symp. on Discrete Algorithms*. ACM/SIAM, 2008.

[3] E. Elkind, A. Sahai, and K. Steiglitz. Frugality in path auctions. In *Proc. 15th ACM Symp. on Discrete Algorithms*. ACM/SIAM, 2004.

[4] A. Karlin, D. Kempe, and T. Tamir. "beyond vcg: Frugality in truthful mechanisms". In *Proc. 46th IEEE Symp. on Foundations of Computer Science*, 2005.

[5] K. Talwar. The price of truth: Frugality in truthful mechanisms. In *Proc. 20th Annual Symp. on Theoretical Aspects of Computer Science*, 2003.