CS 684 Algorithmic Game Theory	Scribe: Ashwin Machanavajjhala
Instructor: Eva Tardos	April 14, 2004

1 VCG : Vickrey Clarke Groves Mechanism

In today's lecture, we are going to look at the shortest path problem. Consider a graph $G = (V, E)$ and a cost function $c : E \rightarrow \Re$. We want to find the shortest path between two vertices $s, t \in V$. In this setting, we model each edge e having cost c_e as a selfish agent A_e. The *agent cost* is 0 if the edge is not in the selected path and c_e if it is on the selected path. If e is on the selected path, A_e gets a payment of p_e. Since the agent is selfish, the agent A_e tries to maximize $p_e - c_e$.

The social welfare goal is to minimize the total cost of the path chosen; i.e., $\min_{all\ paths\ p} \Sigma_{e \in p} c_e$. Note that money is a vehicle used by the economy to ensure social good. Hence, we want to minimize the true cost of the path and not the payments made in the process of acquiring a path. We can write the social welfare objective $\min_{all\ paths\ p} \Sigma_{e \in p} c_e$ as $c(p)$ where p is the shortest path.

Consider an agent A_e. If the agent gets a payment p_e, the agents goal is to

$$\begin{array}{rll} Maximize & p_e - c_e & \text{if } e \text{ is on the selected path} \\ & p_e & \text{if } e \text{ is not on the selected path} \end{array}$$

Note that if the payment is such that it is independent of the agent cost (which is c_e when e is on the path and 0 when e is not on the path), then the agent will truthfully reveal the real cost of the edge. As a first attempt, if $c(p)$ denotes the true cost of a path, we can have payments:

$$\begin{array}{rll} p_e & = c_e - c(p) & \text{if } e \text{ is on the selected path } p \\ & = -c(p) & \text{if } e \text{ is not on the selected path } p \end{array}$$

In this case, the agent gets a profit of $p_e - c_e = -c(p)$, when e is in the selected path and a profit of $p_e = -c(p)$ when e is not in the selected path p. Though truthful, the above payment scheme does not encourage agents to participate in the path selection. To maintain the truthfulness property and elicit voluntary participation, we could add to a positive bonus b_e to every payment such that b_e is independent of the agent cost. So the payments now become:

$$\begin{array}{rll} p_e & = c_e - c(p) + b_e & \text{if } e \text{ is on the selected path } p \\ & = -c(p) + b_e & \text{if } e \text{ is not on the selected path } p \end{array}$$

We want this b_e to be a value independent of the agent cost for each edge. One such b_e is the true cost of the shortest path p^e avoiding e. Since, p^e is the same as p when e is not on the selected path, we have

$$\begin{array}{rll} p_e & = c_e - c(p) + c(p^e) & \text{if } e \text{ is on the selected path } p \\ & = -c(p) + c(p) & \text{if } e \text{ is not on the selected path } p \end{array}$$

Let us look at this payment scheme. Consider an edge e on the selected path p. By how much can the agent A_e lie about the cost of the edge c_e ? Note that if A_e reports a cost $x_e > c_e + c(p^e) - c(p)$, then e is no longer on the shortest path and will not be chosen. Hence, A_e would report a cost of at most $c_e + c(p^e) - c(p)$. This is exactly the payment made to A_e.

Theorem 1 (Truthfulness) *The shortest path with this payment is truthful.*

Proof. Suppose $e \notin p$. A_e has to report a smaller value for c_e so that a path through c_e becomes the shortest path. Hence, A_e would have a negative benefit.

If $e \in p$, A_e would get all the benefit he can get by being untruthful about c_e even by being truthful about the cost c_e. ■

There are, however, few issues with this solution which have to be addressed to. The first issue is the computation difficulty of the payments. The second issue is that social welfare ignores payments – the sum total of the payments may be much higher than the cost of the shortest path.

Today, we will address the first issue about the computational difficulty of the payments. Note that if the shortest path has k edges, the payments can be computed using k shortest path computations. Suri and Hershberger [1] show that the payments can be computed using only 2 shortest path computations. We outline the ideas in this result next.

First let us consider the special case where the shortest path between s and t passes through all the vertices in the graph. Consider the shortest path between s and t not containing $e = (u, v)$, i.e., $p^{(u,v)}$. For any x which comes before u on the path, $d(s, x)$, which is the shortest distance from s to x is the same whether e is in the graph or not. Similarly, for any y which comes after v in the path, $d(y, t)$ is the same whether G has e or not. p^e should contain an edge in the cut $\{U, V\}$, where $U = \{x | x$ lies on s-u path $\}$ and $V = \{y | y$ lies on v-t path $\}$. Hence, finding the shortest path in $G \setminus e$ reduces to

$$d(s, t;\ G \setminus (u, v) = min_{(x,y) \in cut(U,V) \setminus (u,v)} d(s, x) + c(x, y) + d(y, t)$$

To compute the shortest path between s, t omitting each edge on p, compute the above expression for every edge (u, v) on the path. This can be done efficiently using a Fibonacci heap. At each phase, keep only those edges which are in the current cut in the heap. The minimum extracted from the heap would give that (x, y) which minimized the above equation and hence the new path. This gives a running time of $O(nlogn + m)$ (for analysis see [1]).

In a general undirected graph, not all vertices lie on p. Consider the shortest path tree rooted at s. To find $p^{(u,v)}$, consider the two components V_s and V_t formed by the removing edge (u, v) from the shortest path tree, V_s contains s and V_t contains t. For any $x \in V_s$, $d(s, x)$ is the same even in $G \setminus (u, v)$. However, it is not as obvious that similarly for any $y \in V_t$, $d(y, t)$ is the same even in $G \setminus (u, v)$, since removing the edge (u, v) in the shortest path tree rooted at t might not yield the same partition.

Lemma 2 *Let y be a vertex in component V_t in $G \setminus (u, v)$. Then $d(y, t)$ is the same in G and in $G \setminus (u, v)$.*

Proof. The proof is by contradiction. Suppose the shortest path from y to t contains (u, v), then the path should traverse (u, v) in the direction from u to v. (since the shortest path from v to y is fully contained in V_y). Then this shortest path can be considered as a concatenation of the shortest paths between y, v and between v, t; with the first sub-path containing u. But since $y \in V_y$, the shortest path shows tree rooted at s shows that the shortest path between v, y is completely contained in V_y. Since G is undirected, the shortest path between y, v should just be a reversal of the above path. But one contains u and the other does not contain u, which leads to the required contradiction. ■

We have reduced the problem in the undirected case to the special case where all the vertices are on the shortest path between s and t. So one can use a similar algorithm to solve the problem in undirected graph. When trying to compute $p^{(u,v)}$, consider all the edges between vertices in x, y where $x \in V_s$ and $y \in V_t$ and minimize the cost $d(s,x) + d(x,y) + d(y,t)$.

However, the same kind of reasoning does not seem to hold for directed graphs and it was proved that finding the shortest paths for the vickrey payments requires $\Omega(n(nlogn + m))$ time [2, 3].

References

[1] John Hershberger and Subhash Suri. Vickrey Prices and Shortest Paths: What is an Edge worth? In *Proceedings of the 42nd Symposium on the Foundations of Computer Science, (FOCS 2001).*

[2] John Hershberger and Subhash Suri. Erratum to Vickrey Prices and Shortest Paths: What is an Edge worth? In *Proceedings of the 43nd Symposium on the Foundations of Computer Science, (FOCS 2002).*

[3] John Hershberger and Subhash Suri and Amit Bhosle. On the Difficulty if Some Shortest Path Problems. In *STACS 2003*, pp 343-354.