CS 598CSC: Approximation Algorithms	Lecture date: January 23, 2009
Instructor: Chandra Chekuri	Scribe: Sungjin Im

In the previous lecture, we had a quick overview of several basic aspects of approximation algorithms. We also addressed approximation (both offline and online) algorithms for the Steiner Tree Problem. In this lecture, we explore another important problem – the Traveling Salesperson Problem (TSP).

1 The Traveling Salesperson Problem (TSP)

1.1 TSP in Undirected Graphs

In the Traveling Salesperson Problem, we are given an undirected graph $G = (V, E)$ and cost $c(e) > 0$ for each edge $e \in E$. Our goal is to find a Hamiltonian cycle with minimum cost. A cycle is said to be Hamiltonian if it visits every vertex in V exactly once.

TSP is known to be NP-complete, and so we cannot expect to exactly solve TSP in polynomial time. What is worse, there is no good approximation algorithm for TSP unless $P = NP$. This is because if one can give a good approximation solution to TSP in polynomial time, then we can exactly solve the NP-Complete Hamiltonian cycle problem (HAM) in polynomial time, which is impossible unless $P = NP$. Recall that HAM is the decision problem of deciding whether the given graph $G = (V, E)$ has a Hamiltonian cycle or not.

Theorem 1 ([2]) *The Traveling Salesperson Problem cannot be approximated within any factor, unless $P = NP$.*

Proof: For the sake of contradiction, suppose we have an approximation algorithm $\mathcal{A}$ on TSP with an approximation ratio C. We show a contradiction by showing that using $\mathcal{A}$, we can exactly solve HAM in polynomial time. Let $G = (V, E)$ be the given instance of HAM. We create a new graph $H = (V, E')$ with cost $c(e)$ for each $e \in E'$ such that $c(e) = 1$ if $e \in E$, otherwise $c(e) = B$, where $B = nC + 1$ and $n = |V|$.

We observe that if G has a Hamiltonian cycle, $\mathrm{OPT} = n$, otherwise $\mathrm{OPT} \geq n-1+B = n(C+1)$. (Here, OPT denotes the cost of an optimal TSP solution in H.) Note that there is a "gap" between when G has a Hamiltonian cycle and when it does not. Thus, if $\mathcal{A}$ has an approximation ratio of C, we can tell whether G has a Hamiltonian cycle or not: Simply run $\mathcal{A}$ on the graph H; if G has a Hamiltonian cycle, $\mathcal{A}$ returns a TSP tour in H of cost at most $C\mathrm{OPT} = Cn$. Otherwise, H has no TSP tour of cost less than $n(C + 1)$, and so $\mathcal{A}$ must return a tour of at least this cost. □

Since we cannot solve the general TSP problem, we consider TSP with relaxed conditions. There are two possible ways: allowing visiting each vertex many times or requiring a metric condition on cost on edges. Interestingly, both problems are equivalent; we leave this to readers as an exercise.

We consider the second one which we call *Metric-TSP*. In Metric-TSP, the instance is a complete graph $G = (V, E)$ with cost $c(e)$ on $e \in E$, where c satisfies the triangle inequality, i.e. $c(uw) \leq c(uv) + c(vw)$ for any $u, v, w \in V$.

We first consider a natural greedy approach, the Nearest Neighbor Heuristic (NNH).

NEAREST NEIGHBOR HEURISTIC$(G(V,E), c : E \to \mathcal{R}^+)$:
Start at an arbitrary vertex s,
While (there are unvisited vertices)
  From the current vertex u, go to the nearest unvisited vertex v.
Return to s.

Problems to Work On:

1. Prove that NNH is an $O(\log n)$-approximation algorithm. (**Hint:** Think back to the proof of the $2H_{|S|}$-approximation for the Greedy Steiner Tree Algorithm.)

2. NNH is not an $O(1)$-approximation algorithm; can you find an example to show this?

There are $O(1)$-approximation algorithms for TSP; we now consider an MST-based algorithm. (See Fig. 1.)

TSP-MST$(G(V,E), c : E \to \mathcal{R}^+)$:
Compute an MST T of G.
Obtain an Eulerian graph $H = 2T$ by *doubling* edges of T
An Eulerian tour of $2T$ gives a tour in G.
Obtain a Hamiltonian cycle by shortcutting the tour.

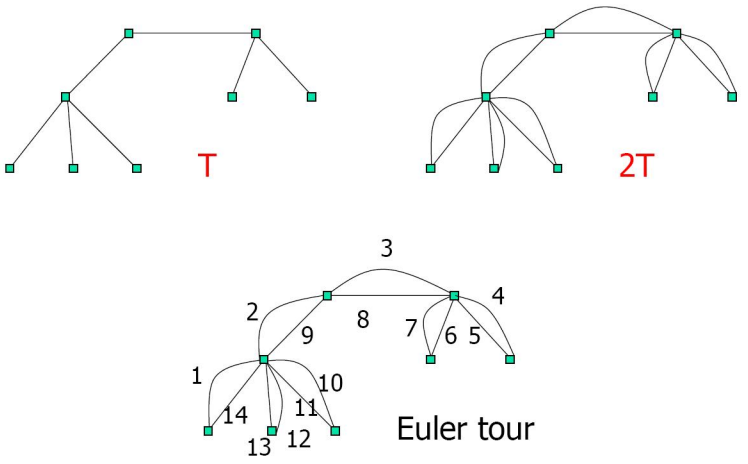

Figure 1: MST Based Heuristic

Recall that graph G has an Eulerian tour if an only if every vertex v of G has even degrees. Thus we can always find an Eulerian graph H by doubling edges of T. This simple algorithm gives a 2-approximation.

Theorem 2 *MST heuristic(TSP-MST) is a 2-approximation algorithm.*

Proof: We have $c(T) = \sum_{e \in E(T)} c(e) \leq \text{OPT}$, since we can get a tree by removing an edge from the optimal Hamiltonian cycle. Thus $c(H) = 2c(T) \leq 2\text{OPT}$. Also shortcutting only decreases the cost. □

We observe that the loss of a factor 2 in the approximation ratio is due to doubling edges; we did this in order to obtain an Eulerian tour. But any graph in which all vertices have even degree is Eulerian, so one can still get an Eulerian tour by adding edges only between odd degree vertices. Christofides Heuristic [1] exploits this and improves the approximation ratio from 2 to 1.5. See Fig. 2 for a snapshot.

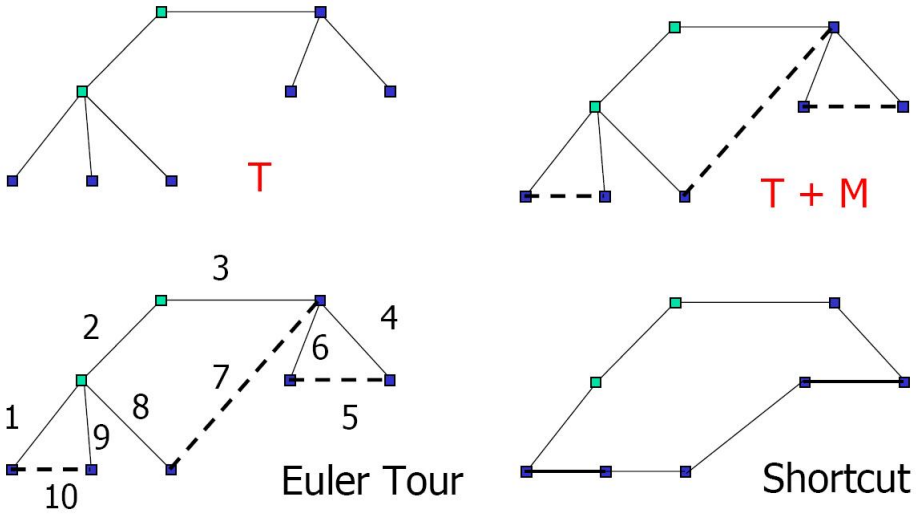

Figure 2: Christofides Heuristic

CHRISTOFIDES HEURISTIC$(G(V,E), c : E \to \mathcal{R}^+)$:
Compute an MST T of G.
Let S be the vertices of odd degree in T. (Note: $\|S\|$ is even)
Find a minimum cost matching M on S in G
Add M to T to obtain an Eulerian graph H.
Compute an Eulerian tour of H.
Obtain a Hamilton cycle by shortcutting the tour.

Theorem 3 *Christofides Heuristic is a 1.5-approximation algorithm.*

Proof: The main part of the proof is to show that $c(M) \leq .5\text{OPT}$. Suppose that $c(M) \leq .5\text{OPT}$. Then, since the solution of Christofides Heuristic is obtained by shortcutting the Eulerian tour on H, its cost is no more than $c(H) = c(T) + c(M) \leq 1.5\text{OPT}$. (Refer to the proof of Lemma 2 for the fact $c(T) \leq \text{OPT}$.) Therefore we focus on proving that $c(M) \leq .5\text{OPT}$.

Let $Tour^*$ be a Hamiltonian tour which achieves OPT. We can get $Tour_S$ on S by shortcutting $Tour^*$. Clearly, $c(Tour_S) \leq \text{OPT}$. Let $v_1, v_2, ..., v_{|S|}, v_{|S|+1} = v_1$ be the order of vertices in $|S|$ visited by $Tour_S$. Recall that $|S|$ is even. Let $M_1 = \{v_1v_2, v_3v_4, ...v_{|S|-1}v_{|S|}\}$ and $M_2 = \{v_2v_3, v_4v_5, ...v_{|S|}v_1\}$. Intuitively, we get M_1 by taking every alternate edge from $Tour_S$ and M_2 by taking the other edges. Note that both M_1 and M_2 are matchings, and $c(M_1)+c(M_2) = c(Tour_S) \leq \text{OPT}$. WLOG, suppose that $c(M_1) \leq c(M_2)$. Then we have $c(M_1) \leq .5\text{OPT}$. Also we know that $c(M) \leq c(M_1)$, since M is a min cost matching on S. Hence we have $c(M) \leq c(M_1) \leq .5\text{OPT}$, which completes the proof. □

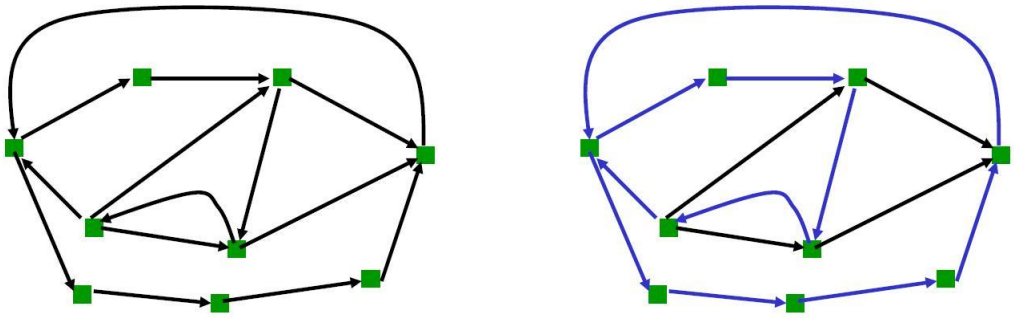

Figure 3: A directed graph and a valid Hamiltonian walk

Notes:

1. In practice, local search heuristics are widely used and they perform extremely well. A popular heuristic *2-Opt* is to swap pairs from xy, zw to xz, yw or xw, yz, if it improves the tour.

2. There have been no improvements to Metric-TSP since Christofides heuristic was discovered in 1976. It remains a major open problem to improve the approximation ratio of $\frac{3}{2}$ for Metric-TSP; it is conjectured that the Held-Karp LP relaxation [3] gives a ratio of $\frac{4}{3}$.

1.2 TSP in Directed Graphs

In this subsection, we consider TSP in directed graphs. As in undirected TSP, we need to relax the problem conditions to get any positive result. Again, allowing each vertex to be visited multiple times is equivalent to imposing the triangle inequality $c(uw) \leq c(uv) + c(vw)$ for all u, v, w. We focus on the first model, which we call asymmetric TSP (ATSP). We are given a directed graph $G = (V, A)$ with cost $c(a) > 0$ for each arc $a \in A$ and our goal is to find a closed walk visiting all vertices. Note that we are allowed to visit each vertex multiple times, as we are looking for a walk, not a cycle. For an example of a valid Hamiltonian walk, see Fig. 3.

It is important to observe that the MST-based heuristic does not work in this setting. This is because costs on edges are not symmetric, and thus doubling edges might blow up the cost by more than a factor of two. Hence, we need another approach. The *Cycle Shrinking Algorithm* repeatedly finds a min-cost cycle cover and shrinks cycles, combining the cycle covers found. Recall that a cycle cover is a collection of disjoint cycles covering all vertices. It is known that finding a minimum cost cycle cover can be done in polynomial time. The Cycle Shrinking Algorithm achieves $\log_2 n$ approximation ratio.

CYCLE SHRINKING ALGORITHM$(G(V, A), c : A \to \mathcal{R}^+)$:
Transform G s.t. G is complete and satisfies $c(uv) + c(vw) \geq c(uv)$ for $\forall u, v, w$
Find a minimum cost cycle cover
Pick a proxy node for each cycle
Recursively solve problem on proxies - extend using cycles

The first step which transforms G so that it satisfies $c(uv) \leq c(uw) + c(wv)$ for $\forall u, v, w$ can be easily done by setting $c(uv)$ to be the cost of the shortest path for each pair $u, v \in V$. Note that

we are implicitly assuming that G is strongly connected. For a snapshot of the Cycle Shrinking Algorithm, see Fig. 4, where cycle covers C_1, C_2 and C_3 are added one by one.

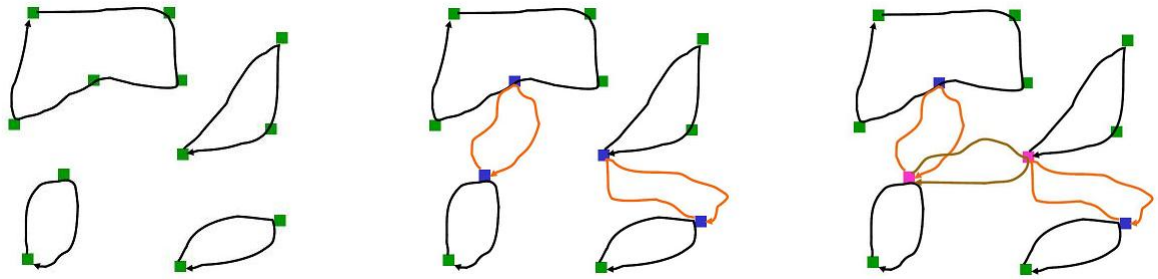

Figure 4: A snapshot of Cycle Shrinking Algorithm. To the left, a cycle cover C_1. In the center, blue vertices indicate proxy nodes, and a cycle cover C_2 is found on the proxy nodes. To the right, pink vertices are new proxy nodes, and a cycle cover C_3 is found on the new proxy nodes.

Theorem 4 *The Cycle Shrinking Algorithm is a* $\log_2 n$*-approximation for ATSP.*

Let $\mathcal{C} = \{C_1, C_2, ..., C_k\}$ be the collection of cycle covers, where C_i is the i_{th} cycle cover found by the algorithm. Let $c(C_i)$ be the cost of C_i. We prove Theorem 4 by showing that the cost of each cycle cover $c(C_i) \leq \text{OPT}$ and the number cycle covers $k \leq \log_2 n$ in Lemma 5 and Lemma 6, respectively. Theorem 4 will directly follow by combining these two lemmas.

Lemma 5 *For each cycle cover* C_i*,* $c(C_i) \leq \text{OPT}$*.*

Proof: Let V_i be the proxy nodes at state i, i.e. $V_i = V(C_i)$. Let G_i be the induced subgraph of G on V_i, i.e. $G_i = G[V_i]$. We observe that (1) $\text{OPT}(V_i) \leq \text{OPT}(G)$, where $\text{OPT}(V_i)$ and $\text{OPT}(G)$ denote the cost of the optimal Hamiltonian walk on V_i and G, respectively. This can be shown by shortcutting the optimal Hamiltonian tour of G so that it passes only V_i. Let W_i be the optimal Hamiltonian walk on V_i. We can show that W_i can be transformed into a cycle cover C_i' only by shortcutting. It can be done by repeatedly shortcutting two remaining arcs uv and vw where v is visited more than one time, until W_i becomes a cycle cover. This keeps the invariant that all vertices are visited and each shortcutting decreases the cost of the tour. Thus we have (2) $c(C_i') \leq \text{OPT}(V_i)$. Finally, we know (3) $c(C_i) \leq c(C_i')$, since C_i is a minimum cost cycle cover on V_i. Combining (1), (2) and (3) completes the proof. □

Lemma 6 *There are at most* $\log_2 n$ *cycle covers in the outcome of Cycle Shrinking Algorithm, i.e.* $k \leq \log_2 n$

Proof: This can be shown by observing that at most half the vertices survive to the next stage, i.e. $|V_{i+1}| \leq 1/2|V_{i+1}|$. □

Notes:

1. The running time of the Cycle Shrinking Algorithm is $O(n^{2.5})$; it is dominated by finding a min cost cycle cover in the first stage. It is known that the running time can be reduced to $O(n^2)$ with a small loss in the approximation ratio.

2. The best approximation ratio so far known is $0.842 \log_2 n$.

3. It has remained an open problem for more than 25 years whether there exists a constant factor approximation for ATSP.

2 Some Definitions

2.1 NP Optimization Problems

In this section, we cover some formal definitions related to approximation algorithms. We start from the definition of optimization problems. Let Π be an optimization problem. Π can be either a min or max type. Instances I of Π are a subset of Σ^*. We denote the set of feasible solutions by $\mathcal{S}(I)$. We assume that for each I and solution $S \in \mathcal{S}(I)$ there is a real/rational number $val(S, I)$. The goal is, given I, to find $\mathrm{OPT}(I) = \min_{S \in \mathcal{S}(I)} val(S, I)$ if Π is a minimization problem.

Now let us formally define NP optimization(NPO) which is the class of optimization problems corresponding to NP.

Definition 7 *Π is in NPO if*

- *given $x \in \Sigma^*$, cwe an check if x is an instance of Π in poly($|x|$) time.*
- *for each I, and $S \in \mathcal{S}(I)$, $|S|$ is poly($|I|$).*
- *there exists a poly-time decision procedure that for each I and $x \in \Sigma^*$, decides if $x \in \mathcal{S}(I)$*
- *$val(I, S)$ is a poly-time computable function*

We observe that for an NPO problem Π, one can always get its decision version $L(\Pi, B) = \{I : \mathrm{OPT}(I) \leq B\}$. It is easy to see that $L(\Pi, B) \in NP$.

2.2 Relative Approximation

When Π is a minimization problem, recall that we say an approximation algorithm $\mathcal{A}$ is said to have approximation ratio α iff

- $\mathcal{A}$ is a polynomial time algorithm
- for all instance I of Π, $\mathcal{A}$ produces a feasible solution $\mathcal{A}(I)$ s.t. $val(\mathcal{A}(I), I) \leq \alpha\, val\,(\mathrm{OPT}(I), I)$. (Note that $\alpha \geq 1$.)

Approximation algorithms for maximization problems are defined similarly. An approximation algorithm $\mathcal{A}$ is said to have approximation ratio α iff

- $\mathcal{A}$ is a polynomial time algorithm
- for all instance I of Π, $\mathcal{A}$ produces a feasible solution $\mathcal{A}(I)$ s.t. $val(\mathcal{A}(I), I) \geq \alpha\, val\,(\mathrm{OPT}(I), I)$. (Note that $\alpha \leq 1$.)

For maximization problems, it is also common to see use $1/\alpha$ (which must be ≥ 1) as approximation ratio.

2.3 Additive Approximation

Note that all the definitions above are about relative approximations; one could also define *additive* approximations. $\mathcal{A}$ is said to be an α-additive approximation algorithm, if for all I, $val(\mathcal{A}(I)) \leq \mathrm{OPT}(I)+\alpha$. Most NPO problems, however, do not allow any additive approximation ratio because $\mathrm{OPT}(I)$ has a scaling property.

To illustrate the scaling property, let us consider Metric-TSP. Given an instance I, let I_β denote the instance obtained by increasing all edge costs by a factor of β. It is easy to observe that for each $S \in \mathcal{S}(I) = \mathcal{S}(I_\beta)$, $val(S, I_\beta) = \beta val(S, I_\beta)$ and $\mathrm{OPT}(I_\beta) = \beta\mathrm{OPT}(I)$. Intuitively, scaling edge by a factor of β scales the value by the same factor β. Thus by choosing β sufficiently large, we can essentially make the additive approximation(or error) negligible.

Lemma 8 *Metric-TSP does not allow any additive approximation algorithm unless $P = NP$.*

Proof: For simplicity, suppose every edge has integer cost. For the sake of contradiction, suppose there exists an additive α approximation $\mathcal{A}$ for Metric-TSP. Given I, we run the algorithm on I_β and let S be the solution, where $\beta = 2\alpha$. We claim that S is the optimal solution for I. We have $val(S, I) = val(S, I_\beta)/\beta \leq \mathrm{OPT}(I_\beta)/\beta + \alpha/\beta = \mathrm{OPT}(I) + 1/2$, as $\mathcal{A}$ is α-additive approximation. Thus we conclude that $\mathrm{OPT}(I) = val(S, I)$, since $\mathrm{OPT}(I) \leq val(S, I)$, and $\mathrm{OPT}(I)$, $val(S, I)$ are integers. This is impossible unless $P = NP$. □

Now let us consider two problems which allow additive approximations. In the Planar Graph Coloring, we are given a planar graph $G = (V, E)$. We are asked to color all vertices of the given graph G such that for any $vw \in E$, v and w have different colors. The goal is to minimize the number of different colors. It is known that to decide if a planar graph admits 3-coloring is NP-complete, while one can always color any planar graph G with using 4 colors. Further, one can efficiently check whether a graph is 2-colorable (that is, if it is bipartite). Thus, the following algorithm is a 1-additive approximation for Planar Graph Coloring: If the graph is bipartite, color it with 2 colors; otherwise, color with 4 colors.

As a second example, consider the Edge Coloring Problem, in which we are asked to color edges of a given graph G with the minimum number of different colors so that no two adjacent edges have different colors. By Vizing's theorem [5], we know that one can color edges with either $\Delta(G)$ or $\Delta(G) + 1$ different colors, where $\Delta(G)$ is the maximum degree of G. Since $\Delta(G)$ is a trivial lower bound on the minimum number, we can say that the Edge Coloring Problem allows a 1-additive approximation. Note that it is known to be NP-complete to decide whether the exact minimum number is $\Delta(G)$ or $\Delta(G) + 1$.

2.4 Hardness of Approximation

Now we move to hardness of approximation.

Definition 9 (Approximability Threshold) *Given a minimization optimization problem Π, it is said that Π has an approximation threshold $\alpha^*(\Pi)$, if for any $\epsilon > 0$, Π allows $\alpha^*(\Pi) + \epsilon$ approximation but does not allow $\alpha^*(\Pi) - \epsilon$ approximation.*

If $\alpha^*(\Pi) = 1$, it implies that Π is solvable in polynomial time. Many NPO problems Π are known to have $\alpha^*(\Pi) > 1$ assuming that $P \neq NP$. We can say that approximation algorithms try to decrease the upper bound on $\alpha^*(\Pi)$, while hardness of approximation attempts to increase lower bounds on $\alpha^*(\Pi)$.

To prove hardness results on NPO problems in terms of approximation, there are largely two approaches; a direct way by reduction from NP-complete problems and an indirect way via gap reductions. Here let us take a quick look at an example using a reduction from an NP-complete problem.

In the (metric) k-center problem, we are given an undirected graph $G = (V, E)$ and an integer k. We are asked to choose k centers from V. The goal is to minimize the maximum distance to a center, i.e. $\min_{S \subseteq V, |S|=k} \max_{v \in V} \mathrm{dist}_G(v, S)$, where $\mathrm{dist}_G(v, S) = \min_{u \in S} \mathrm{dist}_G(u, v)$.

The k-center problem has approximation threshold 2, since there are a few 2-approximation algorithms for k-center and there is no $2-\epsilon$ approximation algorithm for any $\epsilon > 0$ unless $P = NP$. We can prove the inapproximability using a reduction from the Dominating Set Problem. Given an undirected graph $G = (V, E)$, $S \subseteq V$ is said to be a dominating set if any $v \in V$, either $v \in S$ or v is adjacent to some u in S. The goal is to decide whether G has a dominating set in G of size k, i.e. $|S| = k$. This problem is known to be NP-complete.

Theorem 10 ([4]) *Unless $P = NP$, there is no $2 - \epsilon$ approximation for k-center for any $\epsilon > 0$.*

Proof: Let I be an instance of Dominating Set Problem. We create an instance I' of k-center while keeping graph G and k the same. If I has a dominating set of size k then $\mathrm{OPT}(I') = 1$, since every vertex can be reachable from the Dominating Set by at most one hop. Otherwise, we claim that $\mathrm{OPT}(I') \geq 2$. This is because if $\mathrm{OPT}(I') < 2$, then every vertex must be within distance 1, which implies the k-center that witnesses $\mathrm{OPT}(I')$ is a dominating set of I. Therefore, the $(2 - \epsilon)$ approximation for k-center can be used to solve the Dominating Set Problem. This is impossible, unless $P = NP$. □

References

[1] N. Christofides. Worst-case analysis of a new heuristic for the traveling salesman problem. *Technical Report, Graduate School of Industrial Administration, CMU* 1976.

[2] S. Sahni and T.F. Gonzalez. P-Complete Approximation Problems. *J. of the ACM* 23: 555-565, 1976.

[3] M. Held and R. M. Karp. The traveling salesman problem and minimum spanning trees. *Operations Research* 18: 1138–1162, 1970.

[4] W. L. Hsu and G.L. Nemhauser. Easy and hard bottleneck location problems. *Discrete Applied Mathematics*. 1:209-216, 1979.

[5] D. West. *Introductin to Graph Theory*. 2d ed. Prentice Hall. 277-278, 2001.