# Title

Computer Science 604

Advanced Algorithms

Lecture 9a: Approximation Algorithms for TSP

David Juedes

School of EECS

juedes@cs.ohiou.edu

# A Variant of TSP

A natural variant of the Traveling Salesman Problem is to restrict our attention to cost functions that satisfy "reasonable" constraints.

One "reasonable" constraint to restrict our attention to cost functions that satisfy the *triangle inequality*, i.e., that

$$c(u,w) \le c(u,v) + c(v,w)$$

for all $u, v, w \in V$.

Restricting the cost function to satisfy the triangle inequality does not alter the NP-completeness of the traveling salesman problem. (We will call this version of TSP, TSP-TRI.)

However, this restriction allows us to find near optimal tours for the TSP-TRI.

# TSP and MST

Given a connected graph $G = (V, E)$, a set of edges $T$ in $G$ forms a *spanning tree* if the subgraph formed by $T$ is a tree and the set of edges "span" the set vertices in the sense that each vertex in the graph is an endpoint for some edge in $T$.

The cost of a spanning tree is

$$c(T) = \sum_{(u,v) \in T} c(u, v).$$

A natural optimization problem is to find a *minimum cost spanning tree* (MST).

This optimization problem has a polynomial-time solution.

# MST, cont'd

Two well-known algorithms: Prim's algorithm and Kruskal's algorithm. Kruskal's algorithm takes $O(|E| \log |E|)$ steps. Prim's algorithm takes $O(|E| \log |V|)$.

As we will see next, we can find a near optimal solution to TSP-TRI by using MST.

# A near optimal solution to TSP-TRI

**The Algorithm** Input: A complete graph $G = (V, E)$ and a cost function $c$ that satisfies the triangle inequality.

1. Select a vertex $r$ in $V$ as the root vertex.
2. Construct a minimum cost spanning tree $T$ for $G$ by using $r$ as the root of the tree in Prim's algorithm.
3. Let $L$ be a list of vertices in a preorder walk of the tree $T$ starting at the root.
4. Return the Hamiltonian Cycle $H$ that visits the vertices in the order $L$.

# MST, cont'd

Recall that a *preorder traversal* of a tree has the following form.

- Visit the vertex $v$.
- For each child $u$ of $v$, perform preorder-traversal($u$).

**Theorem**: The algorithm given above produces a Hamiltonian tour whose total cost is at most 2 times the cost of the optimal tour.

**Proof**: Let $H^*$ denote an optimal tour for the given set of vertices. Let $H$ be the tour produced by our algorithm. It suffices to show that $c(H) \leq 2c(H^*)$.

## Proof, cont'd

If we delete one edge from the Hamiltonian tour, then the tour becomes a spanning tree. So, let $T$ be a minimum cost spanning tree. Then,

$$c(T) \leq c(H^*).$$

A full walk of $T$ lists the vertices when they were first visited and also whenever they are returned to after a visit to a subtree. Let us call this walk $W$.

**Example**: (See board).

Since the full walk $W$ traverses every edge exactly twice, we have

$$c(W) = 2c(T).$$

## Proof, cont'd

These facts gives us that

$$c(W) \leq 2c(H^*),$$

and so the cost of $W$ is within a factor of 2 of the cost of the optimal tour.

Generally, $W$ is not a tour. However, we can delete a visit to any vertex from $W$ and the cost will not increase because $c$ satisfies the triangle inequality.

That is, if a vertex $v$ is delete from $W$ between visits to $u$ and $w$, the resulting ordering goes directly from $u$ to $w$.

# Proof, cont'd

By repeating this operation, we can remove from $W$ all but the first visit to each vertex. This ordering is the same as the preorder walk. This gives us a Hamiltonian tour $H$. Since $H$ was obtained by delete vertices from the full walk $W$, we have

$$c(H) \leq c(W).$$

Combining the inequalities completes the proof. $\square$

# Christofides's Algorithm for TSP-TRI

We can improve our approximation ratio for TSP-TRI by applying another notion of matching.

Given a graph with $2n$ vertices and weighted edges, a **matching** is a set of $n$ edges such that each vertex has exactly one edge incident on it. A matching with least weight possible is called a **minimum matching**. The minimum matching problem can be solved in $O(n^3)$ on a graph with $n$ vertices. (Gabow, Journal of the ACM (1976), pp. 221–234.)

How is the cost of the minimum weighted matching in a graph related to the cost of the minimum length TSP tour?

(Assume that we have an even number of vertices in the graph.)

# Christofides's Algorithm

Is it clear that if $C$ is the cost of the minimum weighted matching and $H$ is the cost of the minimum TSP tour, then $C \leq 1/2H$.

To see this, notice that if you have a hamiltonian tour in a graph, then we can split the tour into two matchings. (Take every other edge.)

Since the weight of both matchings is at least that of the minimum matching, it follows that $C \leq 1/2H$.

# Christofides's Algorithm

1. Find a Minimum Cost Spanning Tree (MST) in the Graph.

2. Look at the subgraph $G'$ of $G$ containing all vertices in the MST of odd degree. (Notice that there must be an even number of vertices in such a graph.)

3. Find the minimum matching in $G'$. (Notice that the cost of the minimum matching in $G'$ is at most half the cost of the minimum TSP tour in $G$. To see this, construct a tour in $G'$ from the minimum tour in $G$ by skiping vertices not in $G'$. Notice that this tour is at most as long as the original tour. Hence, the cost the minimum weighted matching is at most 1/2 of this cost.)

# Christofides's Algorithm

4. Construct an Eulerian tour from the graph composed of the MST and the minimum matching. (Notice that such a tour must exist since each vertex has even degree. Furthermore, this tour has cost at most 3/2 the cost of optimal.)

5. Construct a tour in the graph by eliminating repeated vertices in the Eulerian tour.

# Christofides's Algorithm

Again, notice the step number 5 does not increase the length of the tour because our graph satisfies the triangle inequality.

Therefore, we have that $A(I) \leq 3/2 * OPT(I)$.