18. Approximate TSP

CPSC 535

Kevin A. Wortman

CALIFORNIA STATE UNIVERSITY FULLERTON

TSP

traveling salesperson problem (TSP)
input: a complete undirected graph $G = (V, E)$ where each edge has weight $w(e) \geq 0$
output: a sequence of vertices H forming a Hamiltonian cycle, minimizing total edge weight

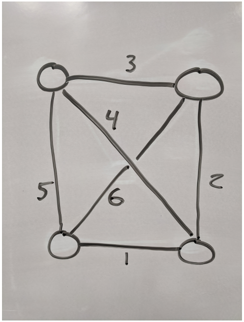
Recall:

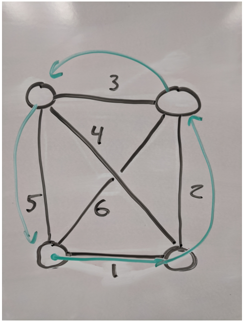
- *Cycle:* path that starts and ends at same vertex
- *Hamiltonian:* visits each vertex exactly once
- every complete graph contains some Hamiltonian cycle

Bad news:

- TSP is NP-complete; if $P \neq NP$, no polynomial-time optimization algorithm
- TSP is also APX-complete; if $P \neq NP$, no PTAS

TSP





Triangle Inequality

Triangle inequality in general: for distance function d and sites a, b, c,

$$d(a, c) \leq d(a, b) + d(b, c)$$

$\Rightarrow$ direct path $a \rightarrow c$ always cheaper than two-step path $a \rightarrow b \rightarrow c$ (or tied)

Triangle inequality in a complete graph: for vertices x, y, z and edge weights w,

$$w(x, z) \leq w(x, y) + w(y, z)$$

$\Rightarrow$ same intuition; adding an intermediate step is never a shortcut
$\Rightarrow$ automatically holds for Euclidean graphs

TSP with Triangle Inequality (TSPTI)

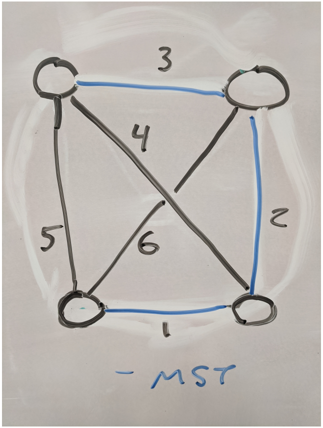
input: a complete undirected graph $G = (V, E)$ where each edge has weight $w(e) \geq 0$; and for any $x, y, z \in V$, $w(x, z) \leq w(x, y) + w(y, z)$
output: (same as conventional TSP)

- **renegotiating** TSP
- different problem; *NP*-completeness and *APX*-completeness proofs may not apply
- less-general problem
- probably still relevant to practical applications of TSP

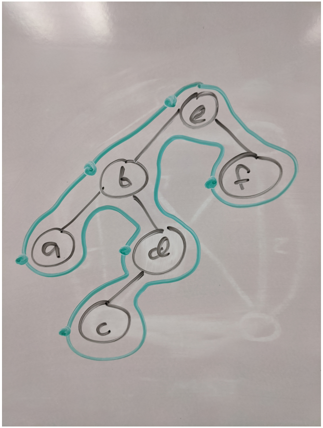
TSPTI Approximation Algorithm Idea

- need a structure that can lower-bound an optimal cycle $H^\star$ and upper-bound our approximate cycle H
- *minimum spanning tree* features
  - minimizes weight of chosen edges
  - connects all vertices
  - can be computed fast
- but an MST is not a Hamiltonian cycle; MST is acyclic, for one thing
- *Euler tour:* cycle around a tree; preorder, inorder, postorder
- build an MST; perform preorder traversal; treat that vertex order as Hamiltonian cycle

Review: Minimum Spanning Trees (MST)



Review: Preorder Traversal

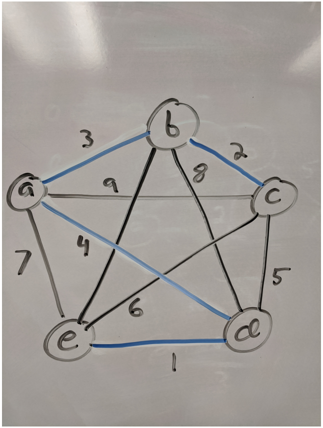


Approximate TSPTI Pseudocode

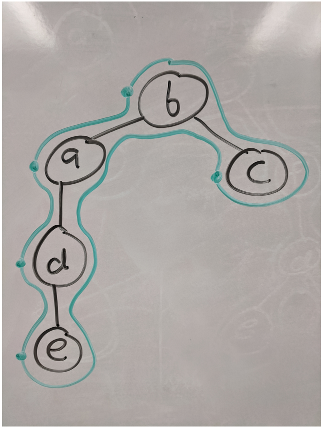
```
function APPROX-TSPTI(G = (V, E), w)
    T = PRIM − MST(G, w)
    H = empty sequence of vertices
    for vertex v in preorder traversal of tree T do
        H.ADDBACK(v)
    end for
    H.ADDBACK(H[0])
    return H
end function
```

Analysis: Prim's algorithm takes $O(m + n \log n)$ (w/ Fibonacci heap), traversal takes $O(m + n)$, total $O(m + n \log n)$ time

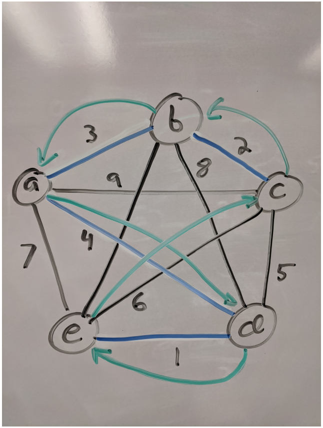
Approximate TSPTI: MST



Approximate TSPTI: Preorder Traversal



Approximate TSPTI: Solution



TSPTI Performance

Lemma: APPROX-TSPTI is a 2-approximation algorithm

Proof Sketch:

- let $H^\star$ be an optimal Hamiltonian cycle for G
- (1) every spanning tree is one edge short of a cycle; and weights are nonnegative; so the weight of our tree T obeys $w(T) \leq w(H^\star)$
- (2) a *full tour* W is the sequence of vertices in both a preorder and postorder tour, and has weight $w(W) = 2w(T)$
- (3) combining (1) and (2), $w(W) \leq 2w(H^\star)$
- (4) our H is like W with some vertices removed, so $w(H) \leq w(W)$
- combining (3) and (4),

$$w(H) \leq w(W) \leq 2w(H^\star)$$

Summary

Vertex cover:

- NP-complete
- exact exhaustive search alg. takes $\Theta(2^n m)$ time
- 2-approximate alg. takes $\Theta(n + m)$ time

TSP:

- NP-complete
- exact exhaustive search alg. takes $\Theta(n!)$ or $\Theta(2^m(n + m))$ time
- 2-approximate algorithm takes $O(m + n \log n)$ time