CS 341: Algorithms
Module 8: Intractability and Undecidability

Armin Jamshidpey, Eugene Zima

Based on lecture notes by many previous CS 341 instructors

David R. Cheriton School of Computer Science, University of Waterloo

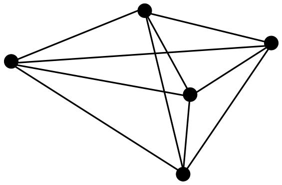
Spring 2019

What to do with NP-hard optimization problems

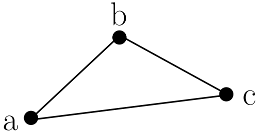
- Efficient exhaustive search (backtracking, branch & bound) $\rightarrow$ exponential time.
- Heuristics

  local search: start with some solution and try to improve it via small " local " changes.
  "simulated annealing " overcomes local optima.
- Approximation algorithms.

Example: TSP for points in the plane with Euclidean distances.



Triangle inequality:



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

Algorithm 1: Approx. Alg.

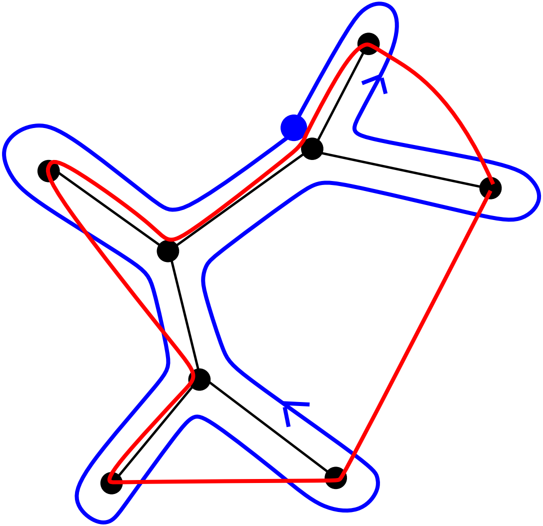
---

1. Compute MST
2. Take a tour by walking around it. (we visit every vertex but maybe more than once)
3. Take short cuts to avoid revisiting.

Note: Triangle inequality $\Rightarrow$ short cuts, shorter.

This can be done in polynomial time.

Example



Let ℓ be the length of the resulting tour

ℓ_{TSP} = length of min TSP tour

claim: $\ell_{\mathrm{TSP}} \leq \ell \leq 2\ell_{\mathrm{TSP}}$.

proof: ℓ_{MST} = length of MST.

$\ell_{\mathrm{MST}} \leq \ell_{\mathrm{TSP}}$, since deleting one edge of TSP gives a spanning tree.

$\ell \leq 2\ell_{\mathrm{MST}}$, since we use every MST edge twice, then take short cuts (use triangle inequality)

Putting these together:

$$\ell \leq 2\ell_{\mathrm{TSP}}.$$

So in polynomial time we find a tour within $2\times$ optimum. We say this algorithm has approximation factor 2.

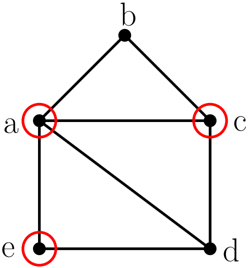
Vertex Cover

For a given graph $G = (V, E)$ find $C \subseteq V$ s.t.

$$(u, v) \in E \Rightarrow u \in C \text{ or } v \in C$$

and $|C|$ is minimum.

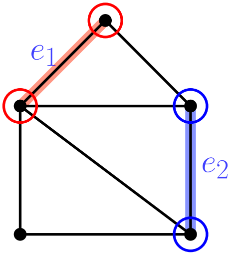
Example:



Algorithm 2: Greedy approximation algorithm

```
C ← ∅
F ← E          // F is uncovered edges.
while F ≠ ∅
        pick (u, v) from F
        add u and v to C
        remove edges incident to u from F
        remove edges incident to v from F
```

Example:



Note that the algorithm takes polynomial time.
Let C =Vertex cover found by the algorithm.
C_{OPT} =a minimum vertex cover.
Claim: $|C| \leq 2 \cdot |C_{\mathrm{OPT}}|$
Proof: The set of edges you pick forms a matching M. Any vertex cover must have at least one vertex from each edge in a matching. $|M| \leq |C_{\mathrm{OPT}}|$. Thus $|C| \leq 2 \cdot |C_{\mathrm{OPT}}|$. This algorithm has approximation factor 2.

General TSP cannot be approximated to within constant factor in polynomial time (unless $P = NP$)

Suppose we have a polynomial time algorithm for TSP that guaranties a tour of length $\leq k \cdot \ell_{\mathrm{TSP}}$.

Claim: We can make a polynomial time algorithm for hamiltonian cycle. Hence $P = NP$.

Algorithm 3: Algorithm for hamiltonian cycle

Input: $G = (V, E)$, $|V| = n$

construct $G' = (V, E' = \{(u, v) : u, v \in V, u \neq v\})$

$$\text{for } e \in E',\ w(e) = \begin{cases} 1 & e \in E \\ k \cdot n & otherwise \end{cases}$$

Run approximation TSP algorithm on G' to get a tour of length ℓ

if $\ell \leq k \cdot n$ output YES

else output NO

Correctness:

In G', a tour that only uses edges of G has length n.
A tour that uses at least one edge not in G has *length* $\geq (n-1) + k \cdot n > k \cdot n$ (assuming $n > 1$).
Claim: $\ell \leq k \cdot n$ iff G has hamiltonian cycle.
Proof: $(\Rightarrow)$ $\ell \leq k \cdot n \Rightarrow \ell = n$ so G has hamiltonian cycle.
$(\Leftarrow)$ G has hamiltonian cycle $\Rightarrow$ G' has a tour of length $n \Rightarrow k-$approx has length $\leq k \cdot n$.