Approximation Algorithms

CPE 349

Theresa Migler-VonDollen

Approximation Algorithms

Definition

Approximation algorithms are algorithms used to find **approximate** solutions to optimization problems.

- Approximation algorithms are often associated to *NP*-hard problems.
  - Why?
- The *approximation ratio* shows how "good" an approximation algorithm is.

Approximation Ratios

Definition

For a minimization problem, a *c-approximation algorithm*, A, is defined to be an algorithm for which the cost, $f(x)$, of the approximate solution, $A(x)$, to an instance x will not be more than a factor c-times the value, OPT, of an optimal solution.
(Note: $c > 1$)

$$OPT \leq f(x) \leq cOPT$$

Approximation Ratios

Definition

For a maximization problem, a *c-approximation algorithm*, A, is defined to be an algorithm for which the cost, $f(x)$, of the approximate solution, $A(x)$, to an instance x will not be less than a factor c-times the value, OPT, of an optimal solution.
(Note: $c < 1$)

$$cOPT \leq f(x) \leq OPT$$

An Approximation Algorithm for Vertex Cover

Vertex Cover

Input: A graph $G = (V, E)$.
Goal: Find a vertex cover of minimum size.

Recall, Vertex Cover is NP-Hard.

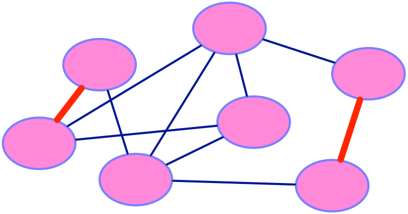
- We developed a dynamic program for the vertex cover problem restricted to trees.
- The problem is NP-Hard, but suppose we need an algorithm for Vertex Cover on general graphs.
- And we need it to be reasonably fast...

An Approximation Algorithm for Vertex Cover

Definition

A *matching* of a graph, $G = (V, E)$, is a subset of edges, $E' \subset$ E, such that no two edges in E' share a vertex.

Example:



An Approximation Algorithm for Vertex Cover

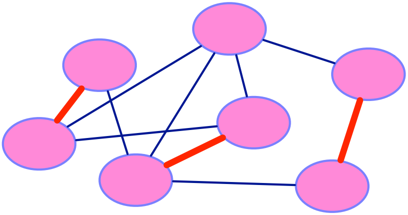
Definition

A *matching* of a graph, $G = (V, E)$, is a subset of edges, $E' \subset$ E, such that no two edges in E' share a vertex.

Definition

A matching is *maximal* if no more edges can be added to the matching.

Example:

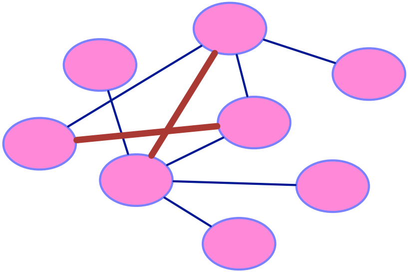


An Approximation Algorithm for Vertex Cover

Definition

A matching is *maximal* if no more edges can be added to the matching.

Note: A maximal matching is not necessarily **maximum**.
Example:



An Approximation Algorithm for Vertex Cover

A maximal matching can be found in polynomial time with a simple greedy algorithm.

- What does this have to do with Vertex Cover?

Relationship between Matchings and Vertex Covers

The number of vertices in any vertex cover must be at least as large as the number of edges in any matching.

- This is because each edge of the matching must be covered by one of its endpoints in any vertex cover.
- Any matching provides a lower bound on OPT.
- Finding such a lower bound is a key step in designing an approximation algorithm:
  - We must compare the quality of the solution found by our algorithm to OPT (which is NP-Hard to compute).

An Approximation Algorithm for Vertex Cover

Relationship between Matchings and Vertex Covers

The number of vertices in any vertex cover must be at least as large as the number of edges in any matching.

- Let S be a set that contains both endpoints of each edge in a maximal matching, M. Then S must be a vertex cover.
  - If S isn't a vertex cover then there must be some edge $e = (u, v) \in E$ for which neither u nor v is in S.
  - Then e could have been added to M (M wasn't maximal).

Two key observations prove our approximation ratio:

1. S, has $2|M|$ vertices.
2. Any vertex cover must have size at least $|M|$.

An Approximation Algorithm for Vertex Cover

Two key observations prove our approximation ratio:

1. S has $2|M|$ vertices.
2. Any vertex cover must have size at least $|M|$.

$$OPT \geq |M|$$

$$2OPT \geq 2|M| = S$$

Approximation Ratio

$$OPT \leq S \leq 2OPT$$

We have a 2-approximation to vertex cover.

An Approximation Algorithm for Vertex Cover

A 2-Approximation Algorithm for Vertex Cover

Input: A graph G.
Output: A vertex cover that is within 2 of the optimal vertex cover.

1: Find a maximal matching of G, $M \subseteq E$
2: Return S, all endpoints of edges in M.

Summary:

- We have no way of finding the optimal vertex cover in polynomial time.
- We can easily find another structure, a maximal matching, with two key properties:
    1. Its size gives us a lower bound on the optimal vertex cover.
    2. It can be used to build a vertex cover, whose size can be related to that of the optimal cover.

An Approximation Algorithm for TSP

Recall the Traveling Salesperson Problem:

Definition

A *Hamiltonian path* is a path in a graph that visits each vertex exactly once.
A *Hamiltonian cycle* is a Hamiltonian path that is a cycle.

Traveling Salesperson Problem

Input: A complete weighted graph.
Goal: Return a Hamiltonian cycle with smallest weight.

This problem is NP-Hard.
We will create an approximation algorithm for this problem restricted to the case where the triangle inequality holds.

An Approximation Algorithm for TSP

Recall the Minimum Spanning Tree Problem.

- How did we solve it?
- How fast were our algorithms?

What does this have to do with TSP?

Relationship between MSTs and TSP

If an edge is removed from the optimal TSP tour, then it becomes a **Spanning Tree**.

$$OPT \geq OPT - \text{one edge} \ \geq MST$$

An Approximation Algorithm for TSP

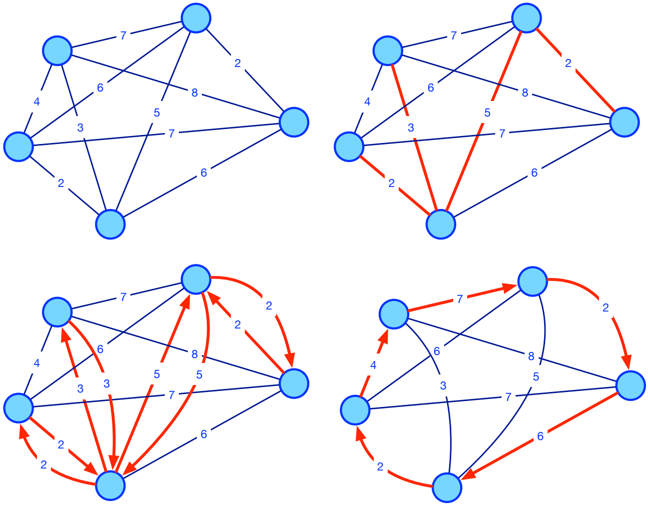
$$OPT \geq OPT - \text{one edge} \geq MST$$

But an MST isn't a TSP tour. Adding an edge to the MST doesn't necessarily make it a TSP tour...

- "Double" every edge and travel along the cycle.
  - This might not give a tour, we may visit some vertices more than once.
  - We bypass already visited cities.
    - This only makes the cost go down because of the triangle inequality.

An Approximation Algorithm for TSP

Example:



An Approximation Algorithm for TSP

A 2-Approximation Algorithm for TSP

Input: A complete weighted graph G.
Output: A Hamiltonian cycle with total weight less than 2 times the weight of the smallest weight Hamiltonian cycle.

1: Find a Minimum Spanning Tree, T, of G
2: "Double" all of the edges in T, for a cycle, C.
3: If any vertex is visited twice, "bypass" it in C.
4: Return C.

Why is this a 2-approximation algorithm?

An Approximation Algorithm for TSP

Why is this a 2-approximation algorithm?

$$T \leq OPT$$

$$C \leq 2T$$

Approximation Ratio

$$OPT \leq C \leq 2OPT$$

Polynomial Time Approximation Scheme

The best approximation algorithm that one could hope to develop is a *PTAS* or *polynomial time approximation scheme*.

Definition

A *PTAS* is an algorithm that takes as an input an instance of an optimization (minimization) problem and a parameter ϵ and in polynomial time produces a solution that is within $1 + \epsilon$ of the optimal solution.

- There is a PTAS for TSP when restricted to the Euclidean plane.
- There is a PTAS for vertex cover when restricted to unit disk graphs.