Great Theoretical Ideas In Computer Science
Victor Adamchik CS 15-251
Carnegie Mellon University

Approximation Algorithms

P ≠ NP

Plan:

Vertex Cover
Metric TSP
3SAT

Computational hardness

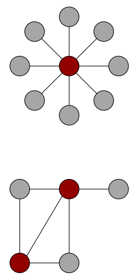
Suppose we are given an NP-complete problem to solve.

Can we develop polynomial-time algorithms that always produce a "good enough" solution?

Vertex cover

Given G=(V,E), find the smallest $S \subseteq V$ s.t. every edge is incident on a vertex in S.

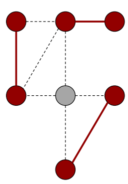
NP-compete problem.



Vertex cover

Lemma. Let M be a matching in G, and S be a vertex cover, then $|S| \geq |M|$.

Proof.
S must cover at least one vertex for each edge in M.

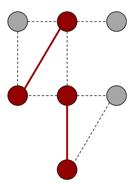


Vertex cover

Def. A matching M is maximal if there is no matching M' such that $M \subseteq M'$.

Which of the following algos. would find a maximal matching:

a) Greedily add edges that are disjoint from the edges added so far, while such edges exist
b) Compute a maximum matching
c) Both
d) Neither



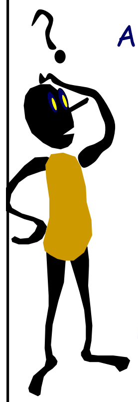
Approximation Vertex Cover

Approx-VC(G):
M ⟵ maximal matching on G
S ⟵ take both endpoints of edges in M
Return S

Theorem. Let OPT(G) be the size of the optimal vertex cover and S = Approx-VC(G). Then $|S| \le 2 \cdot OPT(G)$

Proof. $|S| = 2\,|M| \le 2 \cdot OPT(G)$

Approximation Vertex Cover



Theorem. Let OPT(G) be the size of the optimal vertex cover and S = Approx-VC(G). Then $|S| \le 2 \cdot OPT(G)$

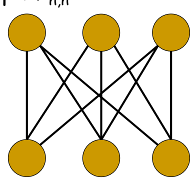
Can we do better than 2 ??

Fact. Nobody knows any algorithm with approximation ratio 1.9

Approximation Vertex Cover

Is 2 a tight bound for this algorithm?

Consider a complete bipartite graph $K_{n,n}$



What is the size of the optimal solution $OPT(K_{n,n})$? **n**

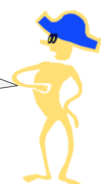
What is the size of any maximal matching $M(K_{n,n})$? **n**

$\text{Approx-VC}(K_{n,n}) = 2n$

Formal Definition

Let P be a minimization problem, and I be an instance of P. Let ALG(I) be a solution returned by an algorithm, and let OPT(I) be an optimal solution. Then ALG(I) is said to be a *c-approximation* algorithm, if for ∀I, $ALG(I) \le c \cdot OPT(I)$.

These notions allow us to circumvent NP-hardness by designing polynomial-time algos with formal worst-case guarantees!



Traveling Salesman Problem

Given a complete undirected graph G=(V,E) with edge cost $c: E \to R^+$, find a min cost Hamiltonian cycle (HC).

Claim: TSP is NP-hard.

Proof by reduction from a HC which is NP-Complete.

Given the input G=(V,E) to HC, we modify it to construct a complete graph G'=(V', E') and cost function as follows:

$c(u,v) = 0$, if edge $(u,v) \in E$
$c(u,v) = 1$, otherwise.

G has a HC iff $|TSP(G')| = 0$

Metric TSP

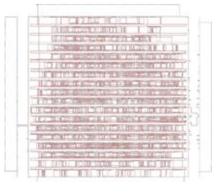
We are allowed to visit vertices multiple times.

We construct a new graph with an edge between every pair of nodes with length equal to the length of the shortest path between them. The shortest path forms a metric:

$$c(u, v) \ge 0,\ \ c(v, v) = 0$$
$$c(u, v) = c(v, u),$$
$$c(u, v) \le c(u, w) + c(w, v)$$

Claim: Metric TSP is NP-hard.

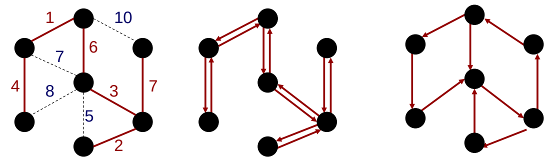
Traveling salesman problem



The largest solved TSP (as of 2013), an 85,900-vertex route calculated in 2006. The graph corresponds to the design of a customized computer chip created at Bell Laboratories, and the solution exhibits the shortest path for a laser to follow as it sculpts the chip.

Approximation Algorithm

Approx-TSP(G):
1) Find a MST of G
2) Complete an Euler tour by doubling edges
3) Remove multiply visited edges (shortcuts)



Approximation Metric-TSP

Theorem. Approx-TSP is a 2-approximation algorithm for a metric TSP.

Proof.

|Approx-TSP| ≤ |Euler Tour| = 2·|MST| ≤ 2·|OPT|

shortcutting decreases the cost.

doubling edges

we can get a spanning tree from HC by removing edges

Christofides Algorithm

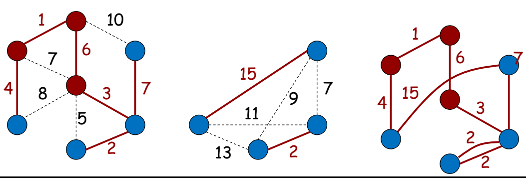
Observe that a factor 2 in the approximation ratio is due to doubling edges; we did this in order to obtain an Eulerian tour.

But any graph with even degrees vertices has an Eulerian tour.

Thus we have to add edges only between odd degree vertices

Christofides Algorithm

Approx-C(G):
T ⟵ MST of G
S ⟵ vertices of odd degree in T
M ⟵ min-cost matching on S
Return: Euler Tour T ∪ M



Christofides Algorithm

Theorem.
Christofides is 3/2 approximation for Metric TSP

The algo has been known for over 30 years and yet no improvements have been made since its discovery.

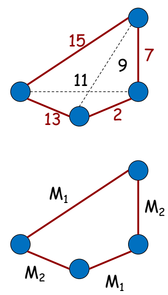
Proof. ALG = c(M) + c(T)

We know that c(T) ≤ OPT.

It remains to show c(M) ≤ $\frac{1}{2}$ OPT.

Christofides Algorithm

Lemma. $c(M) \le \frac{1}{2}$ OPT



Proof. Consider two feasible matching: M_1 and M_2.
Note, $|S|$ is even. Thus,

$c(M) \le \frac{1}{2}(c(M_1) + c(M_2))$

Since $c(M_1) + c(M_2) \le$ OPT

It follows, $c(M) \le \frac{1}{2}$ OPT

Traveling Salesman Problem

Theorem: If $P \neq NP$, then for $\forall c>1$ there is NO a poly-time c-approximation of general TSP.

Proof. To show Ham-cycle $\le_p$ c-approx TSP.

Start with G and create a new complete graph G' with the cost function

$c(u,v) = 1$, if $(u,v) \in E$

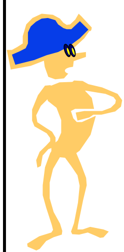
$c(u,v) = c \cdot n$, otherwise (where $n = |V|$)

If G has HC , then $|TSP(G')| = n$.

If G has no HC , then $|TSP(G')| \ge (n-1) + c \cdot n \ge c \cdot n$

Since the $|TSP|$ differs by a factor c, our approx. algorithm can be able to distinguish between two cases, thus decide if G has a ham-cycle.

MAX-SAT



Given a CNF formula (like in SAT), try to maximize the number of clauses satisfied.

CNF is a conjunction of clauses, where each clause is a disjunction of literals $(X_1 \vee X_2 \vee \ldots \vee X_k)$.

Famous NP-complete problem.

Exactly-3-SAT Approximation

Theorem. If every clause has size exactly 3, then there is a simple randomized algorithm that can satisfy at least a 7/8 fraction of clauses.

Proof. Try a random assignment to the variables.

Pr[clause is false] = ?

Since there is only one out of 8 combinations that can make it false, the probability of the clause being false is 1/8.

Exactly-3-SAT Approximation

Theorem. If every clause has size exactly 3, then there is a simple randomized algorithm that can satisfy at least a 7/8 fraction of clauses.

Proof. (cont)

So if there are m clauses total, the expected number satisfied is $(7/8)\, m$.

If the assignment satisfies less, just repeat.

With high probability it won't take too many tries before you do at least as well as the expectation.

Exactly-3-SAT Approximation

With high probability it won't take too many tries before you do at least as well as the expectation.

Proof. (cont)

Let Z be the random variable denoting the number of clauses satisfied by a random assignment.

Let $p_k = \Pr[Z = k]$

$$\le \left(\tfrac{7}{8}m - \tfrac{1}{8}\right) \sum_{0 \le k < 7/8m} p_k \qquad \le m \sum_{7/8m \le k \le m} p_k = p\, m$$

$$E[Z] = \frac{7}{8} m = \sum_{0 \le k \le m} k\, p_k = \sum_{0 \le k < 7/8m} k\, p_k + \sum_{7/8m \le k \le m} k\, p_k \le \left(\tfrac{7}{8}m - \tfrac{1}{8}\right) \cdot 1 + p\, m$$

It follows, $p \ge \frac{1}{8\, m}$

p is the probability that a random assignment satisfies at least 7/8m clauses.

Exactly-3-SAT Approximation

Theorem. If every clause has size exactly 3, then there is a simple randomized algorithm that can satisfy at least a 7/8 fraction of clauses.

Theorem (Hastad, 1997).
If there is an *c-approximation* with $c > 7/8$, then P = NP.

Here's What You Need to Know…

Approximation Algorithms for:

- Vertex Cover
- Metric TSP
- 3SAT