http://www.cs.unh.edu/~ruml/cs758

Coping with NP-Completeness

- find tractable special case
- run only on small inputs
- heuristic optimal algorithm that's usually fast
- heuristic non-optimal algorithm that's always fast
  - if bounded suboptimality: 'approximation algorithm'

Approximation

Approximation Ratio

- Coping with NPC

Approximation

- Approximation
- Vertex Cover
- Proof
- Metric TSP
- Break

Non-approximability

$\rho(n)$-approximation iff cost C for optimal cost C^* is bounded as $\max(C/C^*, C^*/C) \leq \rho(n)$

polynomial-time approximation scheme (PTAS) if, given ϵ as an input parameter, algorithm is a $(1+\epsilon)$-approximation algorithm and polynomial time in input size n

fully-polynomial-time approximation scheme (FPTAS) if running time is polynomial in n and $1/\epsilon$

Vertex Cover

cover all edges using fewest vertices

Vertex Cover

cover all edges using fewest vertices

1. $C \leftarrow \emptyset$, $E' \leftarrow E$
2. while E' is not empty
3.   pick arbitary edge (u, v) from E'
4.   add u and v to C
5.   remove any other edges that touch u or v from E'
6. return C

clearly a cover and polytime. quality vs optimal?

Proof of 2-Approximation

For each (u, v) edge picked, we choose both vertices. No subsequent edge we pick will be adjacent to these vertices. The optimal solution must contain at least one vertex from every edge we pick.

In other words, $|C| = 2|\text{picked}|$ and $|\text{picked}| \leq |C^*|$.

So $|C| \leq 2|C^*|$.

Metric TSP

Cheapest tour (Hamiltonian cycle) over all vertices. Distances satisfy the triangle inequality: $c(u, w) \leq c(u, v) + c(v, w)$.

Metric TSP

Cheapest tour (Hamiltonian cycle) over all vertices. Distances satisfy the triangle inequality: $c(u, w) \leq c(u, v) + c(v, w)$.

1. compute minimum spanning tree
2. construct tour by preorder walk of tree

Clearly a tour and polytime. quality vs optimal?

Metric TSP

Cheapest tour (Hamiltonian cycle) over all vertices. Distances satisfy the triangle inequality: $c(u,w) \leq c(u,v) + c(v,w)$.

1. compute minimum spanning tree
2. construct tour by preorder walk of tree

Clearly a tour and polytime. quality vs optimal?

Proof of 2-Approximation:

1. cost of MST $\leq$ optimal because deleting edge from an optimal tour is a spanning tree
2. if tour really followed edges of MST, would traverse each edge twice, ie, be twice the cost of MST
3. some edges are short-cuts over previously-visited vertices and hence shorter (by triangle inequality)
4. solution $\leq$ twice MST $\leq$ twice optimal

Break

- asst 14

Non-approximability

General TSP

Cheapest tour (Hamiltonian cycle) over all vertices. Distances can be anything.

If P$\neq$NP, no polytime ρ-approximation algorithm exists for TSP.

General TSP

Cheapest tour (Hamiltonian cycle) over all vertices. Distances can be anything.

If P$\neq$NP, no polytime ρ-approximation algorithm exists for TSP.

Show via reduction from Hamiltonian cycle, ie, given ρ-approx alg for TSP, we could decide Hamiltonian cycle.

1. Given G, construct complete graph G' for TSP using edges of cost 1 for edges $\in E$ and cost $\rho|V| + 1$ for all others.
2. If graph contains Hamiltonian cycle, optimal tour has length $|V|$.
3. Any other tour has cost $\geq |V| - 1 + \rho|V| + 1 = |V| + \rho|V|$.
4. Approx alg must return Hamiltonian cycle if it exists. Therefore we can decide Hamiltonian cycle.

MAX 3-CNF SAT

maximize the number of satisfied clauses

2-approximation:

MAX 3-CNF SAT

maximize the number of satisfied clauses

2-approximation: all true or all false!

kinda 8/7-approximation:

MAX 3-CNF SAT

maximize the number of satisfied clauses

2-approximation: all true or all false!

kinda 8/7-approximation: set each variable randomly! (either expected, or guaranteed with expected poly time)

The PCP theorem implies that there exists an $\epsilon > 0$ such that $(1 + \epsilon)$-approximation of MAX-3SAT is NP-hard.

EOLQs

For example:

- What's still confusing?
- What question didn't you get to ask today?
- What would you like to hear more about?

Please write down your most pressing question about algorithms and put it in the box on your way out.
Thanks!