Introduction to Approximation Algorithms

Arijit Bishnu
(arijit@isical.ac.in)

Advanced Computing and Microelectronics Unit
Indian Statistical Institute
Kolkata 700108, India.

Organization

Outline

Approximation ratio

Definition

Approximation ratio

Definition

- An optimal solution $\mathrm{OPT}_\pi(I)$ for an instance I of a minimization [maximization] problem π is a feasible solution that achieves the smallest [largest] objective function value. We would write OPT or $\mathrm{OPT}(I)$ instead of $\mathrm{OPT}_\pi(I)$.

Approximation ratio

Definition

- An optimal solution $\mathrm{OPT}_\pi(I)$ for an instance I of a minimization [maximization] problem π is a feasible solution that achieves the smallest [largest] objective function value. We would write OPT or $\mathrm{OPT}(I)$ instead of $\mathrm{OPT}_\pi(I)$.
- Let π be a minimization [maximization] problem, and let δ be a function, $\delta : \mathbb{Z}^+ \to \mathbb{Q}^+$ with $\delta \geq 1$ [$\delta \leq 1$].

Approximation ratio

Definition

- An optimal solution $\mathrm{OPT}_\pi(I)$ for an instance I of a minimization [maximization] problem π is a feasible solution that achieves the smallest [largest] objective function value. We would write OPT or $\mathrm{OPT}(I)$ instead of $\mathrm{OPT}_\pi(I)$.
- Let π be a minimization [maximization] problem, and let δ be a function, $\delta : \mathbb{Z}^+ \to \mathbb{Q}^+$ with $\delta \geq 1$ [$\delta \leq 1$].
- An algorithm $\mathcal{A}$ is said to be a factor δ approximation algorithm for π if, on each instance I, $\mathcal{A}$ produces a feasible solution s for I such that

Approximation ratio

Definition

- An optimal solution $\mathrm{OPT}_\pi(I)$ for an instance I of a minimization [maximization] problem π is a feasible solution that achieves the smallest [largest] objective function value. We would write OPT or $\mathrm{OPT}(I)$ instead of $\mathrm{OPT}_\pi(I)$.
- Let π be a minimization [maximization] problem, and let δ be a function, $\delta : \mathbb{Z}^+ \to \mathbb{Q}^+$ with $\delta \geq 1$ [$\delta \leq 1$].
- An algorithm $\mathcal{A}$ is said to be a factor δ approximation algorithm for π if, on each instance I, $\mathcal{A}$ produces a feasible solution s for I such that
- $f_\pi(I, s) \leq \delta(|I|) \cdot \mathrm{OPT}(I)$ [$f_\pi(I, s) \geq \delta(|I|) \cdot \mathrm{OPT}(I)$].

Approximation ratio

Definition

- An optimal solution $\mathrm{OPT}_\pi(I)$ for an instance I of a minimization [maximization] problem π is a feasible solution that achieves the smallest [largest] objective function value. We would write OPT or $\mathrm{OPT}(I)$ instead of $\mathrm{OPT}_\pi(I)$.
- Let π be a minimization [maximization] problem, and let δ be a function, $\delta : \mathbb{Z}^+ \to \mathbb{Q}^+$ with $\delta \geq 1$ [$\delta \leq 1$].
- An algorithm $\mathcal{A}$ is said to be a factor δ approximation algorithm for π if, on each instance I, $\mathcal{A}$ produces a feasible solution s for I such that
- $f_\pi(I, s) \leq \delta(|I|) \cdot \mathrm{OPT}(I)$ [$f_\pi(I, s) \geq \delta(|I|) \cdot \mathrm{OPT}(I)$].
- The running time of $\mathcal{A}$ is bounded by a fixed polynomial in $|I|$.

Lower bounding idea for designing approximation algorithm

Lower bounding OPT

Lower bounding idea for designing approximation algorithm

Lower bounding OPT

- We have to fix a bound for δ, i.e. $\frac{f_\pi(I,s)}{OPT} \leq \delta$, but we know nothing of OPT.

Lower bounding idea for designing approximation algorithm

Lower bounding OPT

- We have to fix a bound for δ, i.e. $\frac{f_\pi(I,s)}{OPT} \leq \delta$, but we know nothing of OPT.
- The technique is to lower bound OPT; i.e. find a k_1 such that OPT $\geq k_1$ and find a k_2 such that $f_\pi(I,s) \leq k_2$. Then, $\frac{f_\pi(I,s)}{OPT} \leq \frac{k_2}{k_1} = \delta$.

Approximation algorithm for Vertex Cover

Some background preparation: Matching

Approximation algorithm for Vertex Cover

Some background preparation: Matching

- Given a graph $G = (V, E)$, a subset of the edges $M \subseteq E$ is said to be a matching if no two edges of M share an endpoint.

Approximation algorithm for Vertex Cover

Some background preparation: Matching

- Given a graph $G = (V, E)$, a subset of the edges $M \subseteq E$ is said to be a matching if no two edges of M share an endpoint.
- A matching of maximum cardinality in G is called a maximum matching, and a matching that is maximal under inclusion is called a maximal matching.

Approximation algorithm for Vertex Cover

Some background preparation: Matching

- Given a graph $G = (V, E)$, a subset of the edges $M \subseteq E$ is said to be a matching if no two edges of M share an endpoint.
- A matching of maximum cardinality in G is called a maximum matching, and a matching that is maximal under inclusion is called a maximal matching.
- A maximal matching can be computed in polynomial time by simply greedily picking edges and removing endpoints of picked edges.

Approximation algorithm for Vertex Cover

Some background preparation: Matching

- Given a graph $G = (V, E)$, a subset of the edges $M \subseteq E$ is said to be a matching if no two edges of M share an endpoint.
- A matching of maximum cardinality in G is called a maximum matching, and a matching that is maximal under inclusion is called a maximal matching.
- A maximal matching can be computed in polynomial time by simply greedily picking edges and removing endpoints of picked edges.

Size of maximal matching in G provides a lower bound

Any vertex cover has to pick at least one endpoint of each matched edge.

Approximation algorithm for Vertex Cover

Algorithm for vertex cover

Algorithm:

Approximation algorithm for Vertex Cover

Algorithm for vertex cover

Algorithm:

- Find a maximal matching M in G.

Approximation algorithm for Vertex Cover

Algorithm for vertex cover

Algorithm:

- Find a maximal matching M in G.
- Output both the endpoints of M in G. Let this set of vertices be V', i.e. $|V'| = 2|M|$.

Approximation algorithm for Vertex Cover

Algorithm for vertex cover

Algorithm:

- Find a maximal matching M in G.
- Output both the endpoints of M in G. Let this set of vertices be V', i.e. $|V'| = 2|M|$.

Proof

Approximation algorithm for Vertex Cover

Algorithm for vertex cover

Algorithm:

- Find a maximal matching M in G.
- Output both the endpoints of M in G. Let this set of vertices be V', i.e. $|V'| = 2|M|$.

Proof

- No edge can be left uncovered, otherwise such an edge could have been added to the matching, contradicting its maximality.

Approximation algorithm for Vertex Cover

Algorithm for vertex cover

Algorithm:

- Find a maximal matching M in G.
- Output both the endpoints of M in G. Let this set of vertices be V', i.e. $|V'| = 2|M|$.

Proof

- No edge can be left uncovered, otherwise such an edge could have been added to the matching, contradicting its maximality.
- Because of the lower bound, $|M| \leq$ OPT.

Approximation algorithm for Vertex Cover

Algorithm for vertex cover

Algorithm:

- Find a maximal matching M in G.
- Output both the endpoints of M in G. Let this set of vertices be V', i.e. $|V'| = 2|M|$.

Proof

- No edge can be left uncovered, otherwise such an edge could have been added to the matching, contradicting its maximality.
- Because of the lower bound, $|M| \leq \text{OPT}$.
- Our algorithm returned a vertex cover of size $2|M|$.

Approximation algorithm for Vertex Cover

Algorithm for vertex cover

Algorithm:

- Find a maximal matching M in G.
- Output both the endpoints of M in G. Let this set of vertices be V', i.e. $|V'| = 2|M|$.

Proof

- No edge can be left uncovered, otherwise such an edge could have been added to the matching, contradicting its maximality.
- Because of the lower bound, $|M| \leq \text{OPT}$.
- Our algorithm returned a vertex cover of size $2|M|$.
- Thus, the approximation ratio is 2.

TSP and Hamiltonian cycle

Hamiltonian (HAM) cycle

Given an undirected graph $G = (V, E)$. does it have a cycle that visits each vertex exactly once?

TSP and Hamiltonian cycle

Hamiltonian (HAM) cycle

Given an undirected graph $G = (V, E)$. does it have a cycle that visits each vertex exactly once?

TSP

Given a complete graph with nonnegative edge weights, find a minimum cost cycle visiting every vertex exactly once.

TSP and Hamiltonian cycle

Hamiltonian (HAM) cycle

Given an undirected graph $G = (V, E)$. does it have a cycle that visits each vertex exactly once?

TSP

Given a complete graph with nonnegative edge weights, find a minimum cost cycle visiting every vertex exactly once.

NP-complete

TSP and Hamiltonian cycle are **NP**-complete problems.

Approximation algorithm for (metric) Traveling Salesman Problem

Inapproximability result for TSP

For any polynomial time computable function $f(n)$, TSP cannot be approximated within a factor of $f(n)$, unless **P**=**NP**.

Approximation algorithm for (metric) Traveling Salesman Problem

Inapproximability result for TSP

For any polynomial time computable function $f(n)$, TSP cannot be approximated within a factor of $f(n)$, unless **P**=**NP**.

Proof

Approximation algorithm for (metric) Traveling Salesman Problem

Inapproximability result for TSP

For any polynomial time computable function $f(n)$, TSP cannot be approximated within a factor of $f(n)$, unless **P**=**NP**.

Proof

- Assume, for a contradiction, there is a $f(n)$-factor approximation algorithm, $\mathcal{A}$ for TSP. We show that $\mathcal{A}$ can be used for deciding the HAM cycle problem.

Approximation algorithm for (metric) Traveling Salesman Problem

Inapproximability result for TSP

For any polynomial time computable function $f(n)$, TSP cannot be approximated within a factor of $f(n)$, unless **P**=**NP**.

Proof

- Assume, for a contradiction, there is a $f(n)$-factor approximation algorithm, $\mathcal{A}$ for TSP. We show that $\mathcal{A}$ can be used for deciding the HAM cycle problem.
- Use a HAM $\leq_P$ TSP as follows:

Approximation algorithm for (metric) Traveling Salesman Problem

Inapproximability result for TSP

For any polynomial time computable function $f(n)$, TSP cannot be approximated within a factor of $f(n)$, unless **P**=**NP**.

Proof

- Assume, for a contradiction, there is a $f(n)$-factor approximation algorithm, $\mathcal{A}$ for TSP. We show that $\mathcal{A}$ can be used for deciding the HAM cycle problem.
- Use a HAM $\leq_P$ TSP as follows:
- Assign a weight of 1 to edges of G and a weight of $f(n) \cdot n$ to nonedges, to obtain G'.

Proof continued

Proof continued

Proof continued

Proof continued

- If G has a HAM cycle, then the corresponding tour in G' has cost n, and $\mathcal{A}$ should return a solution of cost $\leq f(n) \cdot n$.

Proof continued

Proof continued

- If G has a HAM cycle, then the corresponding tour in G' has cost n, and $\mathcal{A}$ should return a solution of cost $\leq f(n) \cdot n$.
- If G has no HAM cycle, any tour in G' must use an edge of cost $f(n) \cdot n$, and therefore, $\mathcal{A}$ should return a solution of cost $> f(n) \cdot n$.

Proof continued

Proof continued

- If G has a HAM cycle, then the corresponding tour in G' has cost n, and $\mathcal{A}$ should return a solution of cost $\leq f(n) \cdot n$.
- If G has no HAM cycle, any tour in G' must use an edge of cost $f(n) \cdot n$, and therefore, $\mathcal{A}$ should return a solution of cost $> f(n) \cdot n$.
- Recall $\mathcal{A}$ runs in polynomial time. Thus, HAM cycle can be decided in polynomial time, and **P**=**NP**.

Approximation algorithm for metric TSP

Metric TSP

Given a complete graph G with nonnegative edge weights that satisfy triangle inequality, find a minimum cost cycle visiting every vertex exactly once.

Metric TSP is **NP**-complete but is approximable.

Approximation algorithm for metric TSP

Metric TSP

Given a complete graph G with nonnegative edge weights that satisfy triangle inequality, find a minimum cost cycle visiting every vertex exactly once.
Metric TSP is **NP**-complete but is approximable.

Approximation algorithm for metric TSP

Approximation algorithm for metric TSP

Metric TSP

Given a complete graph G with nonnegative edge weights that satisfy triangle inequality, find a minimum cost cycle visiting every vertex exactly once.
Metric TSP is **NP**-complete but is approximable.

Approximation algorithm for metric TSP

- Find an MST, T of G.

Approximation algorithm for metric TSP

Metric TSP

Given a complete graph G with nonnegative edge weights that satisfy triangle inequality, find a minimum cost cycle visiting every vertex exactly once.
Metric TSP is **NP**-complete but is approximable.

Approximation algorithm for metric TSP

- Find an MST, T of G.
- Double every edge of T to obtain an Eulerian graph $\mathcal{G}$. (An Eulerian graph is one where all vertices can be visited by traversing each edge exactly once.)

Approximation algorithm for metric TSP

Metric TSP

Given a complete graph G with nonnegative edge weights that satisfy triangle inequality, find a minimum cost cycle visiting every vertex exactly once.
Metric TSP is **NP**-complete but is approximable.

Approximation algorithm for metric TSP

- Find an MST, T of G.
- Double every edge of T to obtain an Eulerian graph $\mathcal{G}$. (An Eulerian graph is one where all vertices can be visited by traversing each edge exactly once.)
- Find an Eulerian tour $\mathcal{T}$ of $\mathcal{G}$.

Approximation algorithm for metric TSP

Metric TSP

Given a complete graph G with nonnegative edge weights that satisfy triangle inequality, find a minimum cost cycle visiting every vertex exactly once.
Metric TSP is **NP**-complete but is approximable.

Approximation algorithm for metric TSP

- Find an MST, T of G.
- Double every edge of T to obtain an Eulerian graph $\mathcal{G}$. (An Eulerian graph is one where all vertices can be visited by traversing each edge exactly once.)
- Find an Eulerian tour $\mathcal{T}$ of $\mathcal{G}$.
- Output the tour that visits vertices of G in order of their first appearance (short cutting)in $\mathcal{T}$. Let $\mathcal{C}$ be this tour.

Approximation algorithm for metric TSP

Approximation ratio for metric TSP is 2

Approximation algorithm for metric TSP

Approximation ratio for metric TSP is 2

- Use lower bound idea again. $\text{cost}(T) \leq \text{OPT}$ as deleting any edge from an optimal solution to TSP gives us a spanning tree.

Approximation algorithm for metric TSP

Approximation ratio for metric TSP is 2

- Use lower bound idea again. $\text{cost}(T) \leq \text{OPT}$ as deleting any edge from an optimal solution to TSP gives us a spanning tree.
- $\mathcal{T}$ contains each edge of T twice, so $\text{cost}(\mathcal{T}) = 2 \cdot \text{cost}(T)$.

Approximation algorithm for metric TSP

Approximation ratio for metric TSP is 2

- Use lower bound idea again. $\text{cost}(T) \leq \text{OPT}$ as deleting any edge from an optimal solution to TSP gives us a spanning tree.
- $\mathcal{T}$ contains each edge of T twice, so $\text{cost}(\mathcal{T}) = 2 \cdot \text{cost}(T)$.
- Because of triangle inequality, after the short cutting step, $\text{cost}(\mathcal{C}) \leq \text{cost}(\mathcal{T})$.

Approximation algorithm for metric TSP

Approximation ratio for metric TSP is 2

- Use lower bound idea again. $\text{cost}(T) \leq \text{OPT}$ as deleting any edge from an optimal solution to TSP gives us a spanning tree.
- $\mathcal{T}$ contains each edge of T twice, so $\text{cost}(\mathcal{T}) = 2 \cdot \text{cost}(T)$.
- Because of triangle inequality, after the short cutting step, $\text{cost}(\mathcal{C}) \leq \text{cost}(\mathcal{T})$.
- Combining the above, we get $\text{cost}(\mathcal{C}) \leq 2 \cdot \text{OPT}$.

At Last!!!

Thank you