Complexity Theory

Jan Křetínský

Chair for Foundations of Software Reliability
and Theoretical Computer Science
Technical University of Munich

Summer 2016

Based on slides by Jörg Kreiker

Lecture 18

Approximation

Approximations

Goal

- decision $\to$ optimization
- formal definition of approximation
- hardness of approximation

Plan

- vertex cover: VC
- set cover: SC
- travelling salesman problem: TSP

Planes

Example

Given a set of airports, S, assign gas stations to a smallest subset, C, where planes can cover at most two legs without re-filling.

Formal model

- airports $\sim$ nodes in a graph
- legs $\sim$ undirected edges
- find a smallest set of nodes that covers all edges
- important problem in networks

Vertex Cover

Definition (Cover)

Let $G = (V, E)$ be an undirected graph. A set $C \subseteq V$ is a cover of S if

$$\forall (u, v) \in E.\ u \in C \ \vee \ v \in C$$

Decision problem

$$\mathrm{VC} = \{\langle G, k\rangle \mid G \text{ has a cover } C \text{ and } |C| \leq k\}$$

Optimization problem Min – VC

- given: $G = (V, E)$ undirected
- find: a minimal cover C

MinVC **is NP-hard**

Observation

- C is a cover iff $V \setminus C$ is an independent set.
- C is a minimal cover iff $V \setminus C$ is a maximal independent set.

Proof

- $\forall (u, v).\ u \in C \vee v \in C$

$\Leftrightarrow\ \forall (u, v).\ u \notin V \setminus C \vee v \notin V \setminus C$

$\Leftrightarrow\ \neg\exists (u, v).\ u \in V \setminus C \wedge v \in V \setminus C$

Some optimization problems

- many decision problems we have seen have optimization versions
- both minimization and maximization
- algorithms return best solution with respect to optimization parameter ρ

Examples

problem	min/max	parameter
3SAT	max	number of satisfiable clauses
Indset	max	size of independent set
VC	min	size of cover

Approximation

Computing precise solutions is often **NP**-hard for decision and optimization.

Instead of optimal solutions, in practice it often suffices to come up with approximations.

Definition (ρ-approximation)

A ρ-approximation for a minimization (maximization) problem with optimal solution O, returns a solution that is $\leq \rho O$ ($\geq \rho O$).

Note: ρ may depend on input size.

VC **approximation algorithm**

1. $C \leftarrow \emptyset$
2. **while** C **not** a cover
3. **pick** $(u, v) \in E$ s.t. $u, v \notin C$
4. $C \leftarrow C \cup \{u, v\}$
5. **return** C

Theorem

Algorithm runs in polynomial time and returns a 2-approximation.

Proof Edges picked contain no common vertices. Optimal vertex cover must contain at least one of the nodes, where the algorithm adds both.

Teams

Example

All your friends belong to one or several teams. You want to invite all of them but team-wise. What is the least number of invitations necessary?

Set Cover

- given: finite set U and a family $\mathcal{F}$ of subsets that covers U: $\bigcup \mathcal{F} \supseteq U$
- find: a smallest family $C \subseteq \mathcal{F}$ that covers U

Set Cover is NP-hard

Proof by reduction from vertex cover.

- let $G = (V, E)$ be an undirected graph
- $f(G) = (E, \mathcal{F})$
- $\mathcal{F} = \{E_v \mid v \in V\}$
- $E_v = \{u \mid (u, v) \in E\}$

Greedy algorithm for SC

1. $C \leftarrow \emptyset$, $U' \leftarrow U$
2. **while** $U' \neq \emptyset$
3. **pick** $S \in \mathcal{F}$ **maximizing** $|S \cap U'|$
4. $C \leftarrow C \cup \{S\}$
5. $U' \leftarrow U' \setminus S$
6. **return** C

- greedy algorithms pick the best local options.
- algorithms runs in polynomial time

Roadmap

Just seen

- vertex cover
- 2-approximation algorithm for VC
- set cover
- approximation algorithm

Up next

- show that algorithm is a $\ln n$ approximation
- show that algorithm is a $\ln |S|$ approximation for largest set S
- TSP

What is the approximation ratio?

Need to compare result returned by algorithm with the unknown optimal solution

Observation If U has a k cover, then every subset of U has a k cover too!

Consequence Each step of greedy algorihm covers at least $1/k$ of the uncovered elements!

First bound: $\ln n$

- let $S_1, \ldots, S_t$ be the sequence of sets picked by algorithm
- let U_i be U' after i stages (uncovered)
- observe: $|U_{i+1}| = |U_i \setminus S_{i+1}| \leq |U_i|(1 - 1/k)$
- hence: $|U_{ik}| \leq |U_0|(1 - 1/k)^{ik} \leq \frac{|U|}{e^i}$
- therefore: $t \leq k \ln(n) + 1$

Note: The bound depends on the input length. We say that the greedy algorithm approximates SC to within a logarithmic factor.

Better bound: $\ln|S|$

Theorem

Greedy algorithm approximates the optimal set cover to within a factor of $H(\max\{|S| \mid S \in \mathcal{F}\})$ *where* $H(n) = \Sigma_{i=1}^{n} \frac{1}{i}$

Proof

- imagine a price to be paid by each team
- at each stage 1 euro has to be paid by newly invited team members, split evenly
- $t \leq$ total amount paid

X for each $S \in \mathcal{F}$ selected by the greedy algorithm the total amount paid by its members is at most $\ln|S|$

$\Rightarrow$ the total amount paid (hence t) is less than $k \cdot \ln|S|$ for the largest S selected

Proof of (X)

For an arbitrary set S at any stage of the algorithm holds:

- if m members are uncovered, the algorithm chooses a subset covering at least m elements

$\Rightarrow$ each will pay $\leq 1/m$

- members pay the most, if they are covered one by one

$\Rightarrow$ harmonic series

Travelling Salesman Problem

Example (TSP)

Given a complete, weighted, undirected graph $G = (V, E)$ with non-negative weights. Find a Hamiltonian cycle of minimal cost.

Theorem

TSP *is* **NP***-hard.*

Proof: Reduce from Hamilton cycle (HC) by giving a large weight to non-edges.

Roadmap

Just seen

- **NP**-hard optimization problems
- approximation to within a certain factor
- complexity of approximation for any factor?

Up next

- approximation algorithm for special case of TSP
- Inapproximability results

Triangle Equality Instance

In practice, TSP is applied on graphs that satisfy the triangle inequality:

$$\forall u, v, w \in V. c(u, v) \leq c(u, w) + c(w, v)$$

Approximation algorithm for such *geographical* graphs

1. find minimum spanning tree T_G for $G = (V, E)$
2. traverse along depth-first search of T_G, jump over visited nodes

- algorithm is polynomial
- 2-approximation
  - $c(T_G) \leq$ minimal tour
  - algorithm traversal costs $2 \cdot c(T_G)$ since jumping over costs at most the sum of traversed edges

Roadmap

Just seen

- special TSP instance with polynomial 2-approximation

Up next

- show it is **NP**-hard to approximate general TSP to within any factor $\rho \geq 1$
- introduce gap version of TSP

gap-TSP

Given a complete, weighted, undirected graph $G = (V, E)$ and some constant $h \geq 1$.

Definition (gap-TSP)

A solution to the gap problem, $\mathrm{gap} - \mathrm{TSP}[|V|, h|V|]$, is an algorithm that return

YES if there exists a Hamiltonian cycle of cost $< |V|$

NO if all Hamiltonian cycles have cost $> h|V|$

For all other cases, it may return either yes or no.

Observation: An efficient h-approximation for TSP decides $\mathrm{gap} - \mathrm{TSP}[C, hC]$ for any C.

gap-TSP is NP-hard

Theorem

For any $h \geq 1$, HC $\leq_p$ gap $-$ TSP$[|V|, h|V|]$

Proof: Like GC $\leq_P$ TSP, where non-edge weights are $h|V|$.

$\Rightarrow$ Approximating TSP to within any factor is **NP**-hard.

What have we learnt?

- some NP-hard decision problems have optimization problems that can be efficiently approximated
  - vertex cover within factor 2
  - set cover within a logarithmic factor
  - geographical travelling salesman problem within factor 2
- some other problems are even NP-hard to approximate, for instance, general TSP
- gap problems are a useful tool to establish inapproximablity

Further Reading

Two books on approximation algorithms

- *Dorit Hochbaum*, Approximation Algorithms for NP-Hard Problems, PWS Publishing.
- *Vijay Vazirani*, Approximation algorithms, Springer.

Lecture Notes

Slides are adapted from a CC course by *Muli Safra*:
`http://www.cs.tau.ac.il/~safra/Complexity/Complexity.htm`