

Graph Theory and Optimization

Computational Complexity (in brief)

Nicolas Nisse

Université Côte d'Azur, Inria, CNRS, I3S, France

October 2018

Outline

- 1 Time-complexity Hierarchy
- 2 3-SAT
- 3 Hamiltonian path/cycle
- 4 Vertex-disjoint paths
- 5 Proper Coloring
- 6 Vertex-Cover
- 7 Approximation algorithms
- 8 Other NP-hard problems

Time-Complexity

very brief introduction

Decision Problem

Input: Instance I **Problem:** does I satisfy Property \mathcal{P} ?**Output:** $solution \in \{Yes, No\}$ **ex:** Is graph G connected? Does G admit a s,d -flow of value $\geq k$?...

How to evaluate if:

- 1 a Problem \mathcal{P} is "difficult" or "easy"?
- 2 an algorithm for solving \mathcal{P} is efficient or not?

(classical) Time-complexity of an algorithm \mathcal{A} for solving \mathcal{P}

Number of **elementary operations** of \mathcal{A} as a function of the **size** n of the instance
running time in the worst case

Definitions of **elementary operations** and **size** depend on:

Context, Data Structure, Units of measure...

Time-Complexity very brief introduction

Assume that Algorithm \mathcal{A} has time-complexity $f(N)$:

in the worst case, \mathcal{A} executes $f(N)$ operations on an instance of size N

For which size of instances is your problem feasible?

Assume 10^{10} operations (e.g., addition of 2 integers on 64 bits) per second
(it is more than current desktops)

Complexity $f(N)$	maximum size N
$\Theta(N)$	10^{11}
$\Theta(N \log N)$	10^{10}
$\Theta(N^2)$	$4 \cdot 10^5$
$\Theta(N^3)$	4600
$\Theta(N^4)$	560
$\Theta(2^N)$	36
$\Theta(N!)$	13

Table: Approximation of maximum size to obtain an answer in 10 seconds

Problems solvable by an algorithm with **polynomial running time** are “easy”

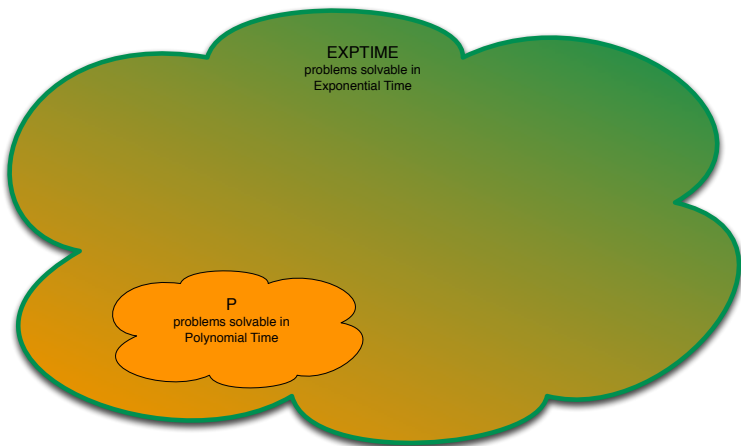
Some polynomial problems

Some examples you may know:

- **Sorting n integers:** $\Theta(n \log n)$ *heap sort, merge sort*
- **Multiplying two $n \times n$ matrices:** $O(n^{2.3728639})$ *[Le Gall, 2014]*
- Decide if m -edge graph is **connected:** $O(m)$ *BFS*
- Compute a **shortest path** in n -node m -edge graph: $O(m + n \log n)$
[Dijkstra]
- Compute **min. spanning tree** in m -edge graph: $O(m \log m)$ *[Kruskal...]*
- **max. flow** and **min. cut** in n -node m -edge graph and max capacity c_{max} :
 $O(m \cdot n \cdot c_{max})$ *[Ford-Fulkerson]*
- **Maximum matching** in n -node graph: $O(n^4)$ *[Edmonds 1965]*
 $O(mn^2)$ *[Micali, Vazirani, 1980]*

All these problems can be solved in time **polynomial** in the size of the input
 \Rightarrow "generally", they are said "easy"

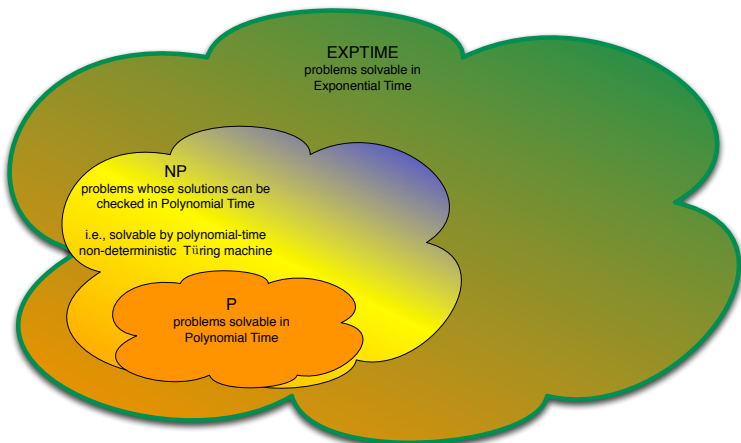
Complexity Hierarchy (very informal and partial description)



EXPTIME: set of the (decision) problems solvable in exponential-time

P: set of the (decision) problems solvable in polynomial-time $P \subset EXPTIME$

Complexity Hierarchy (very informal and partial description)

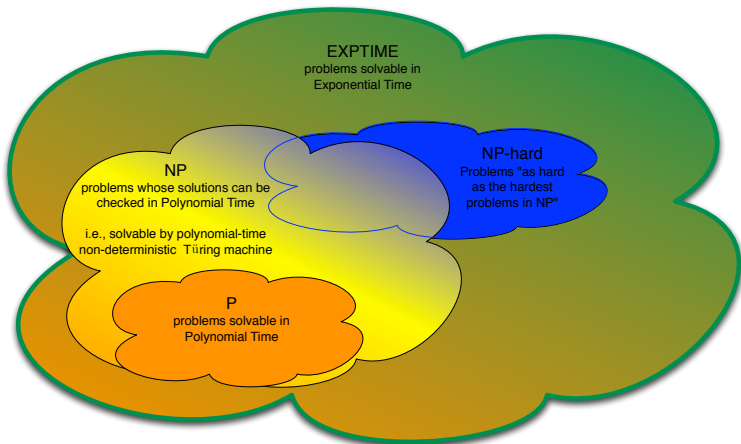


Non-deterministic Polynomial (*NP*): problems that can be solved in polynomial-time by a Non-deterministic Turing machine

$$P \subseteq NP$$

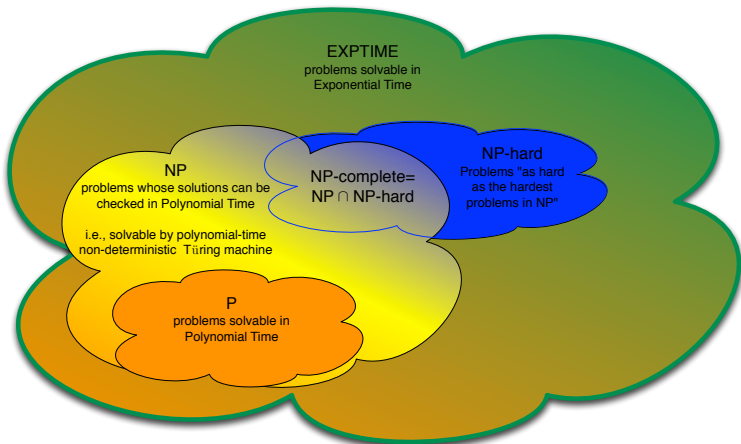
Equivalently, a solution can be checked in polynomial-time

Complexity Hierarchy (very informal and partial description)



NP-hard: problems that are as "difficult" as the "hardest" problems in NP
Solving one of them in polynomial-time would prove that $P = NP$

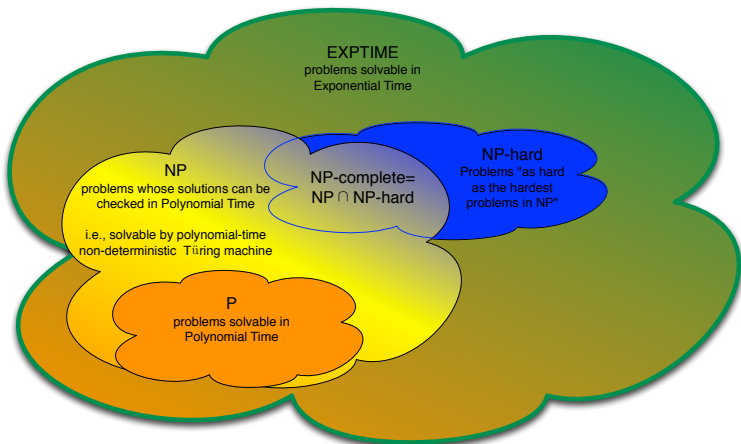
Complexity Hierarchy (very informal and partial description)



NP-hard: problems that are as "difficult" as the "hardest" problems in NP

NP-complete = $NP\text{-hard} \cap NP$

Complexity Hierarchy (very informal and partial description)



NP-hard problems: we do not know if they can be solved in polynomial-time
Roughly, best existing (known) algorithms (generally) enumerate all possible solutions and take a best one

Outline

- 1 Time-complexity Hierarchy
- 2 3-SAT
- 3 Hamiltonian path/cycle
- 4 Vertex-disjoint paths
- 5 Proper Coloring
- 6 Vertex-Cover
- 7 Approximation algorithms
- 8 Other NP-hard problems

Famous NP-complete problems

3-SAT

SAT = Satisfiability

3-SAT is NP-complete

Cook-Levin Theorem (1971)

3-SAT: Given 3-CNF formula $\Phi(v_1, \dots, v_n)$ on n Boolean variables $\exists?$ a Boolean assignment $a: \{v_1, \dots, v_n\} \rightarrow \{0, 1\}$ that satisfies Φ ?

3-CNF= Conjunctive Normal Form, i.e., conjunction of clauses, where a clause is a disjunction of 3 literals.

Ex: $\Phi(a, b, c, d, e) = (a \vee \bar{b} \vee c) \wedge (\bar{e} \vee b \vee \bar{c}) \wedge (\bar{a} \vee \bar{c} \vee e)$ here, the assignment $(a, b, c, d, e) = (1, 1, 0, 0, 1)$ satisfies Φ .

example of algorithm for 3-SAT

Try all the 2^n possible assignments

Remarks: 3-SAT is the first problem to be proved NP-hard.

It is often used to prove that other problems are NP-hard.

Famous NP-complete problems

3-SAT

SAT = Satisfiability

3-SAT is NP-complete

Cook-Levin Theorem (1971)

3-SAT: Given 3-CNF formula $\Phi(v_1, \dots, v_n)$ on n Boolean variables $\exists?$ a Boolean assignment $a: \{v_1, \dots, v_n\} \rightarrow \{0, 1\}$ that satisfies Φ ?

3-CNF= Conjunctive Normal Form, i.e., conjunction of clauses, where a clause is a disjunction of 3 literals.

Ex: $\Phi(a, b, c, d, e) = (a \vee \bar{b} \vee c) \wedge (\bar{e} \vee b \vee \bar{c}) \wedge (\bar{a} \vee \bar{c} \vee e)$ here, the assignment $(a, b, c, d, e) = (1, 1, 0, 0, 1)$ satisfies Φ .

example of algorithm for 3-SAT

Try all the 2^n possible assignments**Remarks:** 3-SAT is the first problem to be proved NP-hard.

It is often used to prove that other problems are NP-hard.

Outline

- 1 Time-complexity Hierarchy
- 2 3-SAT
- 3 Hamiltonian path/cycle
- 4 Vertex-disjoint paths
- 5 Proper Coloring
- 6 Vertex-Cover
- 7 Approximation algorithms
- 8 Other NP-hard problems

Famous NP-c problems

Hamiltonian Path/Cycle

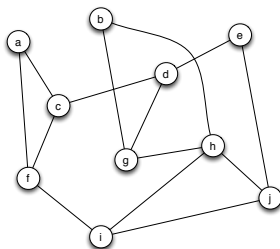
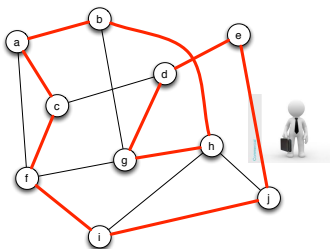
Hamiltonian Path: a spanning path P in a graph G

Hamiltonian Cycle: a spanning cycle C in G

Hamiltonian Path/Cycle is *NP*-complete

[Garey, Johnson]

Hamiltonian path/cycle: Given a graph $G = (V, E)$ with n vertices,
 $\exists?$ an Hamiltonian path/cycle in $G?$



Application: Travelling Salesman Problem (TSP): want to visit all cities, minimizing the distance he has to cross

Exercise: show that the right graph has no Hamiltonian cycle.

Famous NP-c problems

Hamiltonian Path/Cycle

Hamiltonian Path: a spanning path P in a graph G

Hamiltonian Cycle: a spanning cycle C in G

Hamiltonian Path/Cycle is *NP*-complete [Garey, Johnson]

Hamiltonian path/cycle: Given a graph $G = (V, E)$ with n vertices,
 $\exists?$ an Hamiltonian path/cycle in $G?$

Ex of algorithm: Try all the $n!$ orderings of the vertices

Famous NP-c problems

Hamiltonian Path/Cycle

Hamiltonian Path: a spanning path P in a graph G

Hamiltonian Cycle: a spanning cycle C in G

Hamiltonian Path/Cycle is *NP*-complete [Garey, Johnson]

Hamiltonian path/cycle: Given a graph $G = (V, E)$ with n vertices,
 $\exists?$ an Hamiltonian path/cycle in $G?$

Ex of algorithm: Try all the $n!$ orderings of the vertices

Longest path/cycle

Exercise: Let $G = (V, E)$ be a graph and $k \in \mathbb{N}$

Prove that deciding if G has a path/cycle of length $\geq k$ is NP-complete

Famous NP-c problems

Hamiltonian Path/Cycle

Hamiltonian Cycle: cycle that passes through each vertex (exactly once)

Remark: Problems that “look similar” may be “very different”

Tour \approx “cycle” where vertices may be repeated, but not edges.

Eulerian Tour: Tour that passes through each edge (exactly once)

Euler (1736)

Exercise: Prove that deciding if G admits an Eulerian tour is in P

hint: prove that G admits an Eulerian tour \Leftrightarrow each vertex has even degree

Outline

- 1 Time-complexity Hierarchy
- 2 3-SAT
- 3 Hamiltonian path/cycle
- 4 Vertex-disjoint paths
- 5 Proper Coloring
- 6 Vertex-Cover
- 7 Approximation algorithms
- 8 Other NP-hard problems

Famous NP-c problems

Disjoint paths (multi-flow)

Exercise: Let $G = (V, E)$ be a graph, $S, D \subseteq V$, $k \in \mathbb{N}$

Deciding if it exists k vertex-disjoint paths from S to D is in P

hint: use flow algorithm

Disjoint paths is *NP*-complete

[Garey, Johnson]

disjoint paths: Given $G = (V, E)$, $\{s_1, \dots, s_k\} \subseteq V$ and $\{d_1, \dots, d_k\} \subseteq V$
 $\exists? (P_1, \dots, P_k)$ pairwise vertex-disjoint paths s.t. P_i path from s_i to d_i

Remark: can be solved in time $f(k)poly(n)$, i.e., in P if k is fixed

[Robertson and Seymour, 1995]

Famous NP-c problems

Disjoint paths (multi-flow)

Exercise: Let $G = (V, E)$ be a graph, $S, D \subseteq V$, $k \in \mathbb{N}$

Deciding if it exists k vertex-disjoint paths from S to D is in P

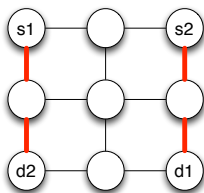
hint: use flow algorithm

Disjoint paths is *NP*-complete

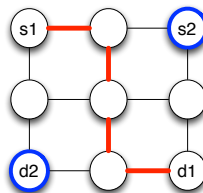
[Garey, Johnson]

disjoint paths: Given $G = (V, E)$, $\{s_1, \dots, s_k\} \subseteq V$ and $\{d_1, \dots, d_k\} \subseteq V$

$\exists? (P_1, \dots, P_k)$ pairwise vertex-disjoint paths s.t. P_i path from s_i to d_i



2 disjoint paths from $\{s_1, s_2\}$ to $\{d_1, d_2\}$



no 2 disjoint paths from s_1 to d_1
and from s_2 to d_2

Remark: can be solved in time $f(k) \text{poly}(n)$, i.e., in P if k is fixed

Famous NP-c problems

Disjoint paths (multi-flow)

Disjoint paths is *NP*-complete

[Garey, Johnson]

disjoint paths: Given $G = (V, E)$, $\{s_1, \dots, s_k\} \subseteq V$ and $\{d_1, \dots, d_k\} \subseteq V$
 $\exists? (P_1, \dots, P_k)$ pairwise vertex-disjoint paths s.t. P_i path from s_i to d_i

Remark: can be solved in time $f(k)poly(n)$, i.e., in *P* if *k* is fixed

[Robertson and Seymour, 1995]

Outline

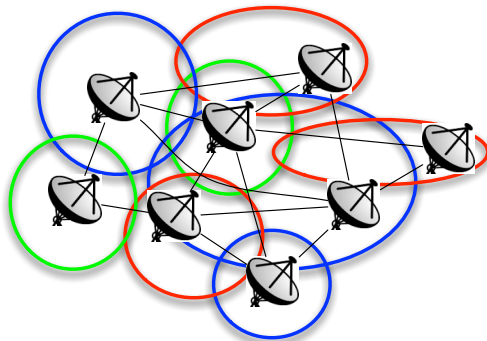
- 1 Time-complexity Hierarchy
- 2 3-SAT
- 3 Hamiltonian path/cycle
- 4 Vertex-disjoint paths
- 5 Proper Coloring
- 6 Vertex-Cover
- 7 Approximation algorithms
- 8 Other NP-hard problems

Famous NP-complete problems

Coloring

Let $G = (V, E)$ be a graph

k -Proper coloring: $c : V \rightarrow \{1, \dots, k\}$ s.t. $c(u) \neq c(v)$ for all $\{u, v\} \in E$.
color the vertices s ($\leq k$ colors) s.t. adjacent vertices receive \neq colors



Vertices = antennas

$\{u, v\} \in E$ iff transmissions of u and v overlap

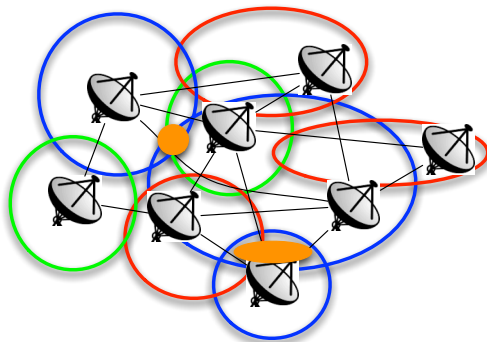
Color = frequency of transmission

Famous NP-complete problems

Coloring

Let $G = (V, E)$ be a graph

k -Proper coloring: $c : V \rightarrow \{1, \dots, k\}$ s.t. $c(u) \neq c(v)$ for all $\{u, v\} \in E$.
color the vertices s ($\leq k$ colors) s.t. adjacent vertices receive \neq colors



Vertices = antennas $\{u, v\} \in E$ iff transmissions of u and v overlap

Color = frequency of transmission

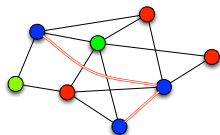
with the coloring in the example: if you live in orange zone \Rightarrow No WiFi!!

Famous NP-complete problems

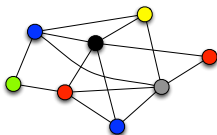
Coloring

Let $G = (V, E)$ be a graph

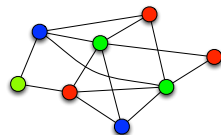
k -Proper coloring: $c : V \rightarrow \{1, \dots, k\}$ s.t. $c(u) \neq c(v)$ for all $\{u, v\} \in E$.
color the vertices $s \leq k$ colors s.t. adjacent vertices receive \neq colors



Unproper 3-coloring (red edges)



Proper 6-coloring



Proper 3-coloring

chromatic number $\chi(G)$: min. k such that G admits a k -Proper coloring.

Exercice: Show that $\chi(G) \leq 2$ if and only if G is bipartite.

Coloring is NP-complete

[Garey, Johnson]

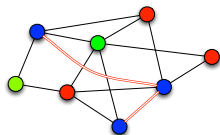
chromatic number: Given $G = (V, E)$ be a graph, $\chi(G) \leq 3?$
 even if G is restricted to be a planar graph

Famous NP-complete problems

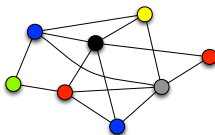
Coloring

Let $G = (V, E)$ be a graph

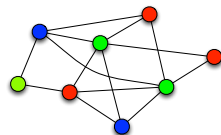
k -Proper coloring: $c : V \rightarrow \{1, \dots, k\}$ s.t. $c(u) \neq c(v)$ for all $\{u, v\} \in E$.
color the vertices $s \leq k$ colors s.t. adjacent vertices receive \neq colors



Unproper 3-coloring (red edges)



Proper 6-coloring



Proper 3-coloring

chromatic number $\chi(G)$: min. k such that G admits a k -Proper coloring.

Exercise: Show that $\chi(G) \leq 2$ if and only if G is bipartite.

Coloring is *NP*-complete

[Garey, Johnson]

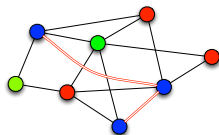
chromatic number: Given $G = (V, E)$ be a graph, $\chi(G) \leq 3?$
 even if G is restricted to be a planar graph

Famous NP-complete problems

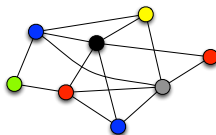
Coloring

Let $G = (V, E)$ be a graph

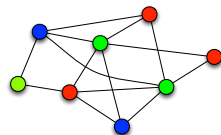
k -Proper coloring: $c : V \rightarrow \{1, \dots, k\}$ s.t. $c(u) \neq c(v)$ for all $\{u, v\} \in E$.
color the vertices s ($\leq k$ colors) s.t. adjacent vertices receive \neq colors



Unproper 3-coloring (red edges)



Proper 6-coloring



Proper 3-coloring

chromatic number $\chi(G)$: min. k such that G admits a k -Proper coloring.

Exercise: Show that $\chi(G) \leq 2$ if and only if G is bipartite.

Coloring is *NP*-complete

[Garey, Johnson]

chromatic number: Given $G = (V, E)$ be a graph, $\chi(G) \leq 3?$
 even if G is restricted to be a planar graph

Outline

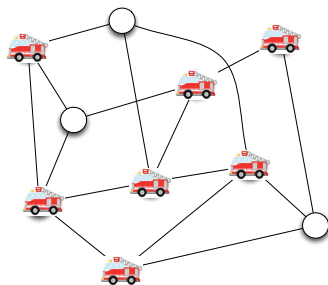
- 1 Time-complexity Hierarchy
- 2 3-SAT
- 3 Hamiltonian path/cycle
- 4 Vertex-disjoint paths
- 5 Proper Coloring
- 6 Vertex-Cover
- 7 Approximation algorithms
- 8 Other NP-hard problems

Famous NP-complete problems

Vertex Cover

Let $G = (V, E)$ be a graph

Vertex Cover: set $K \subseteq V$ such that $\forall e \in E, e \cap K \neq \emptyset$
set of vertices that "touch" every edge



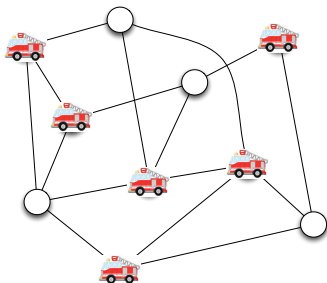
Application: each street must be protected by a fire station

Famous NP-complete problems

Vertex Cover

Let $G = (V, E)$ be a graph

Vertex Cover: set $K \subseteq V$ such that $\forall e \in E, e \cap K \neq \emptyset$
set of vertices that "touch" every edge



Problem: Min. number of fire stations to protect each street?

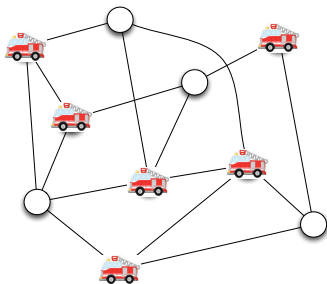
Famous NP-complete problems

Vertex Cover

Let $G = (V, E)$ be a graph

Vertex Cover: set $K \subseteq V$ such that $\forall e \in E, e \cap K \neq \emptyset$

set of vertices that "touch" every edge



Problem: Min. number of fire stations to protect each street?

Min Vertex Cover is *NP*-complete

[Garey, Johnson]

Vertex Cover: Given $G = (V, E)$ be a graph, $k \in \mathbb{N}$,

$\exists? K \subseteq V$ a vertex cover of G such that $|K| \leq k?$

Outline

- 1 Time-complexity Hierarchy
- 2 3-SAT
- 3 Hamiltonian path/cycle
- 4 Vertex-disjoint paths
- 5 Proper Coloring
- 6 Vertex-Cover
- 7 Approximation algorithms
- 8 Other NP-hard problems

How to actually handle NP-hard problems?

NP-hard: No Polynomial-time algorithms known!!!

worst case complexity vs. practice

exponential-time algorithms may be efficient on practical instances

Consider particular instances

Problem \mathcal{P} may be NP-complete in a set \mathcal{I} of instances

but polynomial-time solvable in $\mathcal{I}' \subset \mathcal{I}$

Ex: for any bipartite graph G ,

$$\kappa(G) = \min. \text{ vertex cover}(G) = \max. \text{ matching}(G) = \mu(G)$$

c -Approximation algorithms \mathcal{A} : polynomial-time algorithm s.t.

for any instance I , \mathcal{A} returns a solution with value

for minimization problem: $OPT(I) \leq \text{value}(\mathcal{A}) \leq c \cdot OPT(I)$

for maximization problem: $OPT(I)/c \leq \text{value}(\mathcal{A}) \leq OPT(I)$

Exercise: Give a 2-approximation algorithm for Vertex-Cover

hint: show that, for any graph G , $\mu(G) \leq \kappa(G) \leq 2\mu(G)$

How to actually handle NP-hard problems?

NP-hard: No Polynomial-time algorithms known!!!

worst case complexity vs. practice

exponential-time algorithms may be efficient on practical instances

Consider particular instances

Problem \mathcal{P} may be NP-complete in a set \mathcal{I} of instances

but polynomial-time solvable in $\mathcal{I}' \subset \mathcal{I}$

Ex: for any bipartite graph G ,

$$\kappa(G) = \min. \text{ vertex cover}(G) = \max. \text{ matching}(G) = \mu(G)$$

c-Approximation algorithms \mathcal{A} : polynomial-time algorithm s.t.

for any instance I , \mathcal{A} returns a solution with value

for minimization problem: $OPT(I) \leq \text{value}(\mathcal{A}) \leq c \cdot OPT(I)$

for maximization problem: $OPT(I)/c \leq \text{value}(\mathcal{A}) \leq OPT(I)$

Exercise: Give a 2-approximation algorithm for Vertex-Cover

hint: show that, for any graph G , $\mu(G) \leq \kappa(G) \leq 2\mu(G)$

How to actually handle NP-hard problems?

NP-hard: No Polynomial-time algorithms known!!!

worst case complexity vs. practice

exponential-time algorithms may be efficient on practical instances

Consider particular instances

Problem \mathcal{P} may be NP-complete in a set \mathcal{I} of instances
but polynomial-time solvable in $\mathcal{I}' \subset \mathcal{I}$

Ex: for any bipartite graph G ,

$$\kappa(G) = \min. \text{ vertex cover}(G) = \max. \text{ matching}(G) = \mu(G)$$

c-Approximation algorithms \mathcal{A} : polynomial-time algorithm s.t.

for any instance I , \mathcal{A} returns a solution with value

$$\begin{aligned} \text{for minimization problem: } & \text{OPT}(I) \leq \text{value}(\mathcal{A}) \leq c \cdot \text{OPT}(I) \\ \text{for maximization problem: } & \text{OPT}(I)/c \leq \text{value}(\mathcal{A}) \leq \text{OPT}(I) \end{aligned}$$

Exercise: Give a 2-approximation algorithm for Vertex-Cover

hint: show that, for any graph G , $\mu(G) \leq \kappa(G) \leq 2\mu(G)$

How to actually handle NP-hard problems?

NP-hard: No Polynomial-time algorithms known!!!

worst case complexity vs. practice

exponential-time algorithms may be efficient on practical instances

Consider particular instances

Problem \mathcal{P} may be NP-complete in a set \mathcal{I} of instances
but polynomial-time solvable in $\mathcal{I}' \subset \mathcal{I}$

Ex: for any bipartite graph G ,

$$\kappa(G) = \min. \text{ vertex cover}(G) = \max. \text{ matching}(G) = \mu(G)$$

c-Approximation algorithms \mathcal{A} : polynomial-time algorithm s.t.

for any instance I , \mathcal{A} returns a solution with value

$$\begin{aligned} \text{for minimization problem: } & \text{OPT}(I) \leq \text{value}(\mathcal{A}) \leq c \cdot \text{OPT}(I) \\ \text{for maximization problem: } & \text{OPT}(I)/c \leq \text{value}(\mathcal{A}) \leq \text{OPT}(I) \end{aligned}$$

Exercise: Give a 2-approximation algorithm for Vertex-Cover

$$\text{hint: show that, for any graph } G, \mu(G) \leq \kappa(G) \leq 2\mu(G)$$

Outline

- 1 Time-complexity Hierarchy
- 2 3-SAT
- 3 Hamiltonian path/cycle
- 4 Vertex-disjoint paths
- 5 Proper Coloring
- 6 Vertex-Cover
- 7 Approximation algorithms
- 8 Other NP-hard problems

Some other famous NP-complete problems

Max. Independent set is NP-complete

Independent set: Given a graph $G = (V, E)$, $k \in \mathbb{N}$, $\exists?$ stable set $S \subseteq V$ of size $\geq k$ in G ?

Min. Feedback Vertex Set (FVS) is NP-complete

FVS: Given a digraph $D = (V, A)$, $k \in \mathbb{N}$, $\exists?$ $F \subseteq V$ such that $D \setminus F$ is acyclic?

Min. Set Cover is NP-complete

Set Cover: set E , family of subsets $\mathcal{S} = \{\mathcal{S}_1, \dots, \mathcal{S}_\ell\} \subseteq 2^E$, $k \in \mathbb{N}$
 $\exists?$ $Y \subseteq \mathcal{S}$, $\bigcup_{S \in Y} S = E$, $|\mathcal{S}| \leq k$?

Min. Hitting Set is NP-complete

Hitting Set: set E , family of subsets $\mathcal{S} = \{\mathcal{S}_1, \dots, \mathcal{S}_\ell\} \subseteq 2^E$, $k \in \mathbb{N}$
 $\exists?$ $H \subseteq E$, $H \cap \mathcal{S}_i$ for any $i \leq \ell$, $|H| \leq k$?

Partition is *weakly* NP-complete

Partition: Given set $X = \{x_1, \dots, x_n\}$ of integers,
 $\exists?$ partition (A, B) of X such that $\sum_{x \in A} x = \sum_{x \in B} x$?

Summary: To be remembered

- *P, NP, NP-hard, NP-complete*
- 3-SAT, Hamiltonian path, Coloring, Vertex Cover, ...
- Approximation algorithm