CS 577- Intro to Algorithms

Computational Intractability (Part 4)

Dieter van Melkebeek

December 3, 2020

Outline

Outline

How to handle NP-complete problems

Outline

How to handle NP-complete problems

- Instance structure

Outline

How to handle NP-complete problems

- Instance structure
- Parameter bounds

Outline

How to handle NP-complete problems

- Instance structure
- Parameter bounds
- Approximations

Outline

How to handle NP-complete problems

- Instance structure
- Parameter bounds
- Approximations
- Heuristics

Instance Structure

Instance Structure

Idea

Exploit structure of instances that occur in application setting.

Instance Structure

Idea

Exploit structure of instances that occur in application setting.

Vertex Cover

Instance Structure

Idea

Exploit structure of instances that occur in application setting.

Vertex Cover

Can be solved in polynomial time for

Instance Structure

Idea

Exploit structure of instances that occur in application setting.

Vertex Cover

Can be solved in polynomial time for:

- Trees

Instance Structure

Idea

Exploit structure of instances that occur in application setting.

Vertex Cover

Can be solved in polynomial time for:

- Trees
- Bipartite graphs

Instance Structure

Idea

Exploit structure of instances that occur in application setting.

Vertex Cover

Can be solved in polynomial time for:

- Trees
- Bipartite graphs
- Interval graphs

Instance Structure

Idea

Exploit structure of instances that occur in application setting.

Vertex Cover

Can be solved in polynomial time for:

- Trees
- Bipartite graphs
- Interval graphs
- ...

Parameter Bounds

Parameter Bounds

Idea

Exploit bounds on parameters (other than input size) for instances that occur in application setting.

Parameter Bounds

Idea

Exploit bounds on parameters (other than input size) for instances that occur in application setting.

Vertex Cover

Parameter Bounds

Idea

Exploit bounds on parameters (other than input size) for instances that occur in application setting.

Vertex Cover

Using vertex cover size k as additional parameter

Parameter Bounds

Idea

Exploit bounds on parameters (other than input size) for instances that occur in application setting.

Vertex Cover

Using vertex cover size k as additional parameter:

- Polynomial-time solvable for each fixed k

Parameter Bounds

Idea

Exploit bounds on parameters (other than input size) for instances that occur in application setting.

Vertex Cover

Using vertex cover size k as additional parameter:

- Polynomial-time solvable for each fixed k
  - Exhaustively try all $\binom{n}{k} = \Theta(n^k)$ possible subsets of size k.

Parameter Bounds

Idea

Exploit bounds on parameters (other than input size) for instances that occur in application setting.

Vertex Cover

Using vertex cover size k as additional parameter:

- Polynomial-time solvable for each fixed k
  - Exhaustively try all $\binom{n}{k} = \Theta(n^k)$ possible subsets of size k.
- Fixed-parameter tractable

Parameter Bounds

Idea

Exploit bounds on parameters (other than input size) for instances that occur in application setting.

Vertex Cover

Using vertex cover size k as additional parameter:

- Polynomial-time solvable for each fixed k
  - Exhaustively try all $\binom{n}{k} = \Theta(n^k)$ possible subsets of size k.
- Fixed-parameter tractable
  - Running time $O(2^k \cdot (|V| + |E|))$

Parameter Bounds

Idea

Exploit bounds on parameters (other than input size) for instances that occur in application setting.

Vertex Cover

Using vertex cover size k as additional parameter:

- Polynomial-time solvable for each fixed k
  - Exhaustively try all $\binom{n}{k} = \Theta(n^k)$ possible subsets of size k.
- Fixed-parameter tractable
  - Running time $O(2^k \cdot (|V| + |E|))$
- Kernelization

Parameter Bounds

Idea

Exploit bounds on parameters (other than input size) for instances that occur in application setting.

Vertex Cover

Using vertex cover size k as additional parameter:

- Polynomial-time solvable for each fixed k
  - Exhaustively try all $\binom{n}{k} = \Theta(n^k)$ possible subsets of size k.
- Fixed-parameter tractable
  - Running time $O(2^k \cdot (|V| + |E|))$
- Kernelization
  - Kernel consisting of at most k^2 edges

Fixed-Parameter Tractability

Fixed-Parameter Tractability

Definition

Instances of bit-length n with parameter k can be solved in time $f(k) \cdot n^c$ for some $f : \mathbb{N} \to \mathbb{N}$ and $c \in \mathbb{N}$.

Fixed-Parameter Tractability

Definition

Instances of bit-length n with parameter k can be solved in time $f(k) \cdot n^c$ for some $f : \mathbb{N} \to \mathbb{N}$ and $c \in \mathbb{N}$.

Vertex Cover

Fixed-Parameter Tractability

Definition

Instances of bit-length n with parameter k can be solved in time $f(k) \cdot n^c$ for some $f : \mathbb{N} \to \mathbb{N}$ and $c \in \mathbb{N}$.

Vertex Cover

- Recursive algorithm VC-Decision(V, E, k) based on principle of optimization applied to edge

Fixed-Parameter Tractability

Definition

Instances of bit-length n with parameter k can be solved in time $f(k) \cdot n^c$ for some $f : \mathbb{N} \to \mathbb{N}$ and $c \in \mathbb{N}$.

Vertex Cover

- Recursive algorithm VC-Decision(V, E, k) based on principle of optimization applied to edge
  1. if $E = \emptyset$ then return "yes"
  2. if $k = 0$ then return "no"
  3. pick $e = (u, v) \in E$
  4. return VC-Decision$(V, E \setminus (\{u\} \times V), k - 1) \vee$ VC-Decision$(V, E \setminus (\{v\} \times V), k - 1)$

Fixed-Parameter Tractability

Definition

Instances of bit-length n with parameter k can be solved in time $f(k) \cdot n^c$ for some $f : \mathbb{N} \to \mathbb{N}$ and $c \in \mathbb{N}$.

Vertex Cover

- Recursive algorithm VC-Decision(V, E, k) based on principle of optimization applied to edge
  1. if $E = \emptyset$ then return "yes"
  2. if $k = 0$ then return "no"
  3. pick $e = (u, v) \in E$
  4. return VC-Decision$(V, E \setminus (\{u\} \times V), k-1) \vee$ VC-Decision$(V, E \setminus (\{v\} \times V), k-1)$
- Running time: $O(2^k \cdot (|V| + |E|))$

Kernelization

Kernelization

Definition

Instances of bit-length n with parameter k can be reduced in time n^c to instances of size at most $g(k)$ of the same problem, for some $g : \mathbb{N} \to \mathbb{N}$ and $c \in \mathbb{N}$.

Kernelization

Definition

Instances of bit-length n with parameter k can be reduced in time n^c to instances of size at most $g(k)$ of the same problem, for some $g : \mathbb{N} \to \mathbb{N}$ and $c \in \mathbb{N}$.

Vertex Cover

Kernelization

Definition

Instances of bit-length n with parameter k can be reduced in time n^c to instances of size at most $g(k)$ of the same problem, for some $g : \mathbb{N} \to \mathbb{N}$ and $c \in \mathbb{N}$.

Vertex Cover

- Vertices of degree more than k need to be included.

Kernelization

Definition

Instances of bit-length n with parameter k can be reduced in time n^c to instances of size at most $g(k)$ of the same problem, for some $g : \mathbb{N} \to \mathbb{N}$ and $c \in \mathbb{N}$.

Vertex Cover

- Vertices of degree more than k need to be included.
- A graph G' in which each vertex has degree at most d and has a vertex cover of size s, can have at most $s \cdot d$ edges.

Kernelization

Definition

Instances of bit-length n with parameter k can be reduced in time n^c to instances of size at most $g(k)$ of the same problem, for some $g : \mathbb{N} \to \mathbb{N}$ and $c \in \mathbb{N}$.

Vertex Cover

- Vertices of degree more than k need to be included.
- A graph G' in which each vertex has degree at most d and has a vertex cover of size s, can have at most $s \cdot d$ edges.
- Reduction for VC-Decision:

Kernelization

Definition

Instances of bit-length n with parameter k can be reduced in time n^c to instances of size at most $g(k)$ of the same problem, for some $g : \mathbb{N} \to \mathbb{N}$ and $c \in \mathbb{N}$.

Vertex Cover

- Vertices of degree more than k need to be included.
- A graph G' in which each vertex has degree at most d and has a vertex cover of size s, can have at most $s \cdot d$ edges.
- Reduction for VC-Decision:
  1. $S \leftarrow \{v \in V : \deg(v) > k\}$
  2. $E' \leftarrow E \setminus S \times V$
  3. if $|E'| > (k - |S|) \cdot k$ then return "no"
  4. return decision for $(G', k - |S|)$ where $G' = (V(E'), E')$

Kernelization

Definition

Instances of bit-length n with parameter k can be reduced in time n^c to instances of size at most $g(k)$ of the same problem, for some $g : \mathbb{N} \to \mathbb{N}$ and $c \in \mathbb{N}$.

Vertex Cover

- Vertices of degree more than k need to be included.
- A graph G' in which each vertex has degree at most d and has a vertex cover of size s, can have at most $s \cdot d$ edges.
- Reduction for VC-Decision:
  1. $S \leftarrow \{v \in V : \deg(v) > k\}$
  2. $E' \leftarrow E \setminus S \times V$
  3. if $|E'| > (k - |S|) \cdot k$ then return "no"
  4. return decision for $(G', k - |S|)$ where $G' = (V(E'), E')$
- Reduced instance G' has at most k^2 edges and $2k^2$ vertices.

Approximations

Approximations

Idea

Instead of finding exact optimum, find valid solution whose objective value is close to that of exact optimum.

Approximations

Idea

Instead of finding exact optimum, find valid solution whose objective value is close to that of exact optimum.

Definition

A ρ-approximation algorithm is a polynomial-time algorithm that guarantees closeness to within a multiplicative factor of ρ.

Approximations

Idea

Instead of finding exact optimum, find valid solution whose objective value is close to that of exact optimum.

Definition

A ρ-approximation algorithm is a polynomial-time algorithm that guarantees closeness to within a multiplicative factor of ρ.

Vertex Cover

Approximations

Idea

Instead of finding exact optimum, find valid solution whose objective value is close to that of exact optimum.

Definition

A ρ-approximation algorithm is a polynomial-time algorithm that guarantees closeness to within a multiplicative factor of ρ.

Vertex Cover

Has 2-approximation algorithms:

Approximations

Idea

Instead of finding exact optimum, find valid solution whose objective value is close to that of exact optimum.

Definition

A ρ-approximation algorithm is a polynomial-time algorithm that guarantees closeness to within a multiplicative factor of ρ.

Vertex Cover

Has 2-approximation algorithms:

- Greedy

Approximations

Idea

Instead of finding exact optimum, find valid solution whose objective value is close to that of exact optimum.

Definition

A ρ-approximation algorithm is a polynomial-time algorithm that guarantees closeness to within a multiplicative factor of ρ.

Vertex Cover

Has 2-approximation algorithms:

- Greedy
- Linear programming relaxation

Greedy 2-Approximation for Vertex Cover

Greedy 2-Approximation for Vertex Cover

- Consider maximal matching M in G, i.e., matching that cannot be extended.

Greedy 2-Approximation for Vertex Cover

- Consider maximal matching M in G, i.e., matching that cannot be extended.
- $\mathrm{OPT} \geq |M|$

Greedy 2-Approximation for Vertex Cover

- Consider maximal matching M in G, i.e., matching that cannot be extended.
- $\text{OPT} \geq |M|$
- Let S be set of all endpoints of edges in M.

Greedy 2-Approximation for Vertex Cover

- Consider maximal matching M in G, i.e., matching that cannot be extended.
- $\text{OPT} \geq |M|$
- Let S be set of all endpoints of edges in M.
  - S is a vertex cover.

Greedy 2-Approximation for Vertex Cover

- Consider maximal matching M in G, i.e., matching that cannot be extended.
- $\mathrm{OPT} \geq |M|$
- Let S be set of all endpoints of edges in M.
    - S is a vertex cover.
    - $|S| \leq 2 \cdot |M|$

Greedy 2-Approximation for Vertex Cover

- Consider maximal matching M in G, i.e., matching that cannot be extended.
- $\text{OPT} \geq |M|$
- Let S be set of all endpoints of edges in M.
  - S is a vertex cover.
  - $|S| \leq 2 \cdot |M|$
  - $|S| \leq 2 \cdot |M| \leq 2 \cdot \text{OPT}$

Linear Programming

Linear Programming

- Optimizing a linear objective function over $\mathbb{R}^n$ under linear inequality constraints.

Linear Programming

- Optimizing a linear objective function over $\mathbb{R}^n$ under linear inequality constraints.
- Widely used algorithm: simplex

Linear Programming

- Optimizing a linear objective function over $\mathbb{R}^n$ under linear inequality constraints.
- Widely used algorithm: simplex
- Can be solved in polynomial time.

Linear Programming

- Optimizing a linear objective function over $\mathbb{R}^n$ under linear inequality constraints.
- Widely used algorithm: simplex
- Can be solved in polynomial time.
- No strongly polynomial-time algorithm known.

LP-Based 2-Approximation for Vertex Cover

LP-Based 2-Approximation for Vertex Cover

Integral LP for Vertex Cover

LP-Based 2-Approximation for Vertex Cover

Integral LP for Vertex Cover

- Variables: $x_v \in \mathbb{R}$ for each $v \in V$

LP-Based 2-Approximation for Vertex Cover

Integral LP for Vertex Cover

- Variables: $x_v \in \mathbb{R}$ for each $v \in V$
- Objective: $\min f(x)$ where $f(x) \doteq \sum_{v \in V} x_v$

LP-Based 2-Approximation for Vertex Cover

Integral LP for Vertex Cover

- Variables: $x_v \in \mathbb{R}$ for each $v \in V$
- Objective: $\min f(x)$ where $f(x) \doteq \sum_{v \in V} x_v$
- Constraints:
  - $(\forall\, e = (u, v) \in E)\, x_u + x_v \geq 1$

LP-Based 2-Approximation for Vertex Cover

Integral LP for Vertex Cover

- Variables: $x_v \in \mathbb{R}$ for each $v \in V$
- Objective: min $f(x)$ where $f(x) \doteq \sum_{v \in V} x_v$
- Constraints:
  - $(\forall\, e = (u, v) \in E)\, x_u + x_v \geq 1$
  - $(\forall\, v \in V)\, 0 \leq x_v \leq 1$

LP-Based 2-Approximation for Vertex Cover

Integral LP for Vertex Cover

- Variables: $x_v \in \mathbb{R}$ for each $v \in V$
- Objective: min $f(x)$ where $f(x) \doteq \sum_{v \in V} x_v$
- Constraints:
  - $(\forall\, e = (u, v) \in E)\, x_u + x_v \geq 1$
  - $(\forall\, v \in V)\, 0 \leq x_v \leq 1$
  - All x_v are integral.

LP-Based 2-Approximation for Vertex Cover

Integral LP for Vertex Cover

- Variables: $x_v \in \mathbb{R}$ for each $v \in V$
- Objective: $\min f(x)$ where $f(x) \doteq \sum_{v \in V} x_v$
- Constraints:
  - $(\forall\, e = (u, v) \in E)\, x_u + x_v \geq 1$
  - $(\forall\, v \in V)\, 0 \leq x_v \leq 1$
  - All x_v are integral.

Relaxation

LP-Based 2-Approximation for Vertex Cover

Integral LP for Vertex Cover

- Variables: $x_v \in \mathbb{R}$ for each $v \in V$
- Objective: $\min f(x)$ where $f(x) \doteq \sum_{v \in V} x_v$
- Constraints:
  - $(\forall\, e = (u, v) \in E)\, x_u + x_v \geq 1$
  - $(\forall\, v \in V)\, 0 \leq x_v \leq 1$
  - All x_v are integral.

Relaxation

- Dropping integrality constraints yields genuine LP.

LP-Based 2-Approximation for Vertex Cover

Integral LP for Vertex Cover

- Variables: $x_v \in \mathbb{R}$ for each $v \in V$
- Objective: $\min f(x)$ where $f(x) \doteq \sum_{v \in V} x_v$
- Constraints:
  - $(\forall\, e = (u, v) \in E)\, x_u + x_v \geq 1$
  - $(\forall\, v \in V)\, 0 \leq x_v \leq 1$
  - All x_v are integral.

Relaxation

- Dropping integrality constraints yields genuine LP.
- Find solution of LP: x_v^* for $v \in V$.

LP-Based 2-Approximation for Vertex Cover

Integral LP for Vertex Cover

- Variables: $x_v \in \mathbb{R}$ for each $v \in V$
- Objective: min $f(x)$ where $f(x) \doteq \sum_{v \in V} x_v$
- Constraints:
  - $(\forall\, e = (u, v) \in E)\, x_u + x_v \geq 1$
  - $(\forall\, v \in V)\, 0 \leq x_v \leq 1$
  - All x_v are integral.

Relaxation

- Dropping integrality constraints yields genuine LP.
- Find solution of LP: x_v^* for $v \in V$.
- $f(x^*) \leq \mathrm{OPT}$

LP-Based 2-Approximation for Vertex Cover

Integral LP for Vertex Cover

- Variables: $x_v \in \mathbb{R}$ for each $v \in V$
- Objective: min $f(x)$ where $f(x) \doteq \sum_{v \in V} x_v$
- Constraints:
  - $(\forall\, e = (u, v) \in E)\, x_u + x_v \geq 1$
  - $(\forall\, v \in V)\, 0 \leq x_v \leq 1$
  - All x_v are integral.

Relaxation

- Dropping integrality constraints yields genuine LP.
- Find solution of LP: x_v^* for $v \in V$.
- $f(x^*) \leq \mathrm{OPT}$
- Let $S \doteq \{v \in V : x_v^* \geq 1/2\}$.

LP-Based 2-Approximation for Vertex Cover

Integral LP for Vertex Cover

- Variables: $x_v \in \mathbb{R}$ for each $v \in V$
- Objective: $\min f(x)$ where $f(x) \doteq \sum_{v \in V} x_v$
- Constraints:
  - $(\forall\, e = (u, v) \in E)\, x_u + x_v \geq 1$
  - $(\forall\, v \in V)\, 0 \leq x_v \leq 1$
  - All x_v are integral.

Relaxation

- Dropping integrality constraints yields genuine LP.
- Find solution of LP: x_v^* for $v \in V$.
- $f(x^*) \leq \mathrm{OPT}$
- Let $S \doteq \{v \in V : x_v^* \geq 1/2\}$.
  - S is a vertex cover.

LP-Based 2-Approximation for Vertex Cover

Integral LP for Vertex Cover

- Variables: $x_v \in \mathbb{R}$ for each $v \in V$
- Objective: min $f(x)$ where $f(x) \doteq \sum_{v \in V} x_v$
- Constraints:
  - $(\forall\, e = (u, v) \in E)\, x_u + x_v \geq 1$
  - $(\forall\, v \in V)\, 0 \leq x_v \leq 1$
  - All x_v are integral.

Relaxation

- Dropping integrality constraints yields genuine LP.
- Find solution of LP: x_v^* for $v \in V$.
- $f(x^*) \leq \text{OPT}$
- Let $S \doteq \{v \in V : x_v^* \geq 1/2\}$.
  - S is a vertex cover.
  - $|S| \leq 2 \cdot \sum_{v \in S} x_v^* \leq 2 \cdot \sum_{v \in V} x_v^* = 2 \cdot f(x^*) \leq 2 \cdot \text{OPT}$

Heuristics

Heuristics

- Algorithms that have returned good results in some cases, but no known guarantees.

Heuristics

- Algorithms that have returned good results in some cases, but no known guarantees.
- Often combine local search with restarts to get out of local optimum, using randomness.

Heuristics

- Algorithms that have returned good results in some cases, but no known guarantees.
- Often combine local search with restarts to get out of local optimum, using randomness.
- Often based on physical processes that minimize energy or entropy.

Heuristics

- Algorithms that have returned good results in some cases, but no known guarantees.
- Often combine local search with restarts to get out of local optimum, using randomness.
- Often based on physical processes that minimize energy or entropy.
- Examples: Metropolis, simulated annealing, etc.