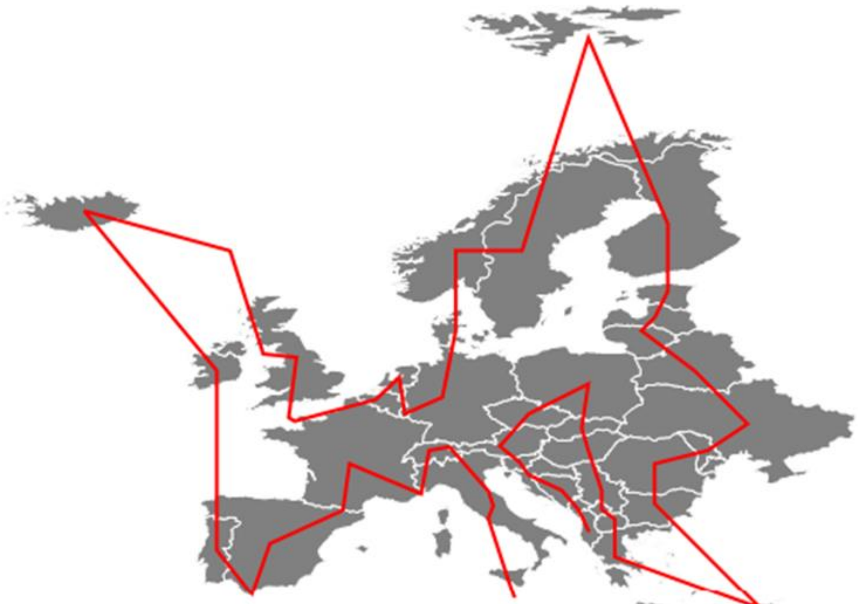
University of Ljubljana, Faculty of Computer and Information Science

Approximation Algorithms



Prof Marko Robnik-Šikonja

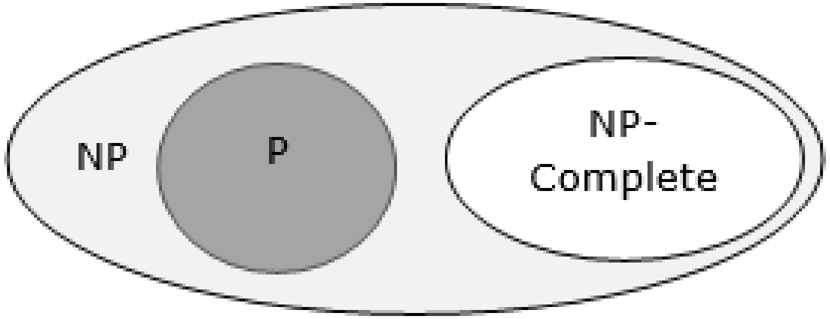
Analysis of Algorithms and Heuristic Problem Solving
Edition 2023

Contents

- refresh your knowledge about NP-completeness in Cormen et al.: Introduction to algorithms, 2009, Chapter 34
- performance ratios
- examples of approximation algorithms
- non-existance of approximation algorithms

- Literature:
  Cormen et al.: Introduction to algorithms, 2009/2022, Chapter 35

NP-completeness refreshment



Next six slides are just a refreshment of important topics in NPC, we require in our course.
Please read Chapter 34 in Cormen et al, Introduction to algorithms, 2009

P and NP problems

- shortest and longest paths
- Euler's trail and Hamilton's cycle
- decision and optimization problems
- problem reductions
- Formal languages: alphabet, language, language operations
- accepting a word, deciding, verification

revision

Definitions

- P={L $\subseteq$ {0,1}*: there exists an algorithm A which decides L in polynomial time}
- verification algorithm A(x, y), where x is an input and y a certificate
- a language is verified with verification algorithm A if for every x $\in$ L there exists a certificate
  L={x$\in${0,1}*: $\exists$ y$\in${0,1}* such that A(x, y) = 1}
- The complexity class NP is a class of languages, verifiable by a polynomial algorithm

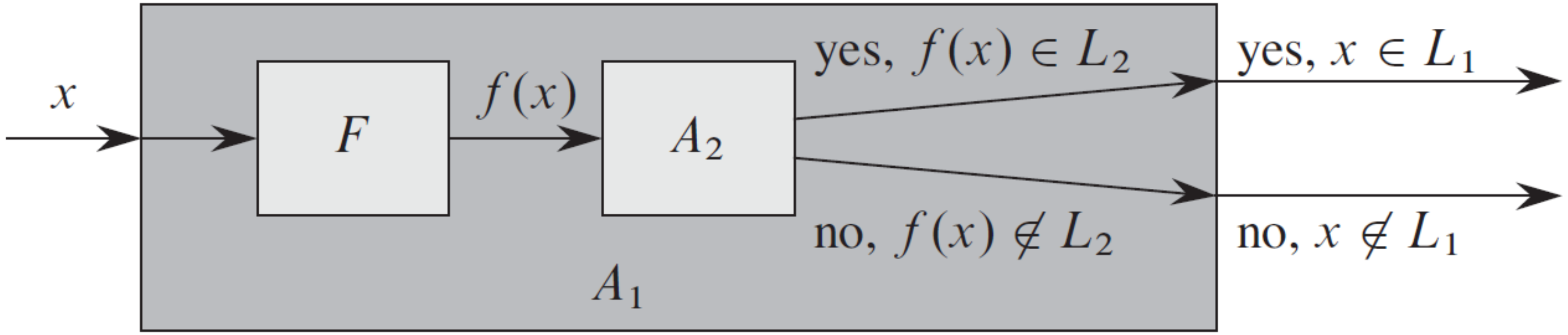
L $\in$ NP $\leftrightarrow$ $\exists$ A(x,y) $\in$ P, constant c, such that
L={x$\in${0,1}*: $\exists$ certificate y: $|y|=O(|x|^c)$
that A(x,y)=1}

revision

Reductions

- Language L_1 is polynomial reducible to language L_2, which we denote as $L_1 \leq_p L_2$, if there exists a polynomially computable function f: $\{0,1\}^* \rightarrow \{0,1\}^*$, such that for all $x \in \{0,1\}^*$ it is true:
  $x \in L_1 \leftrightarrow f(x) \in L_2$

- for languages $L_1, L_2 \subseteq \{0,1\}^*$ and $L_1 \leq_p L_2$ it holds: $L_2 \in P \rightarrow L_1 \in P$



revision

NP-completeness

Language L is NP-complete (NPC) if

1. $L \in NP$
2. $L' \leq_p L$ for each $L' \in NP$

If at least the second point is true, language L is NP-hard.

revision

Proofs of NP-completeness

- If for L is true that $L' \leq_p L$ for some $L' \in NPC$, then L is NP-hard. If $L \in NP$, then $L \in NPC$.

- Proof steps:

1. show $L \in NP$
2. choose a known NP-complete language L'
3. describe reduction algorithm, which computes function f, that for every $x \in L'$ returns $f(x) \in L$
4. prove that for all $x \in \{0,1\}^*$ it holds: $x \in L' \leftrightarrow f(x) \in L$
5. prove that your algorithm computes f in polynomial time

revision

A few well-known NP-complete problems

- CSAT - logical Circuit Satisfiability,
- FSAT - logical Formula Satisfiability,
- 3CNF-SAT - formula in 3-Conjunctive Normal Form Satisfiability
- CLIQUE - existence of Cliques in a graph,
- VERTEX COVER – a minimal set of vertices that cover all the edges of a graph
- HAM Hamiltonian cycle of a graph,
- TSP Travelling Salesman Problem,
- SUBSET-SUM – the subset of numbers equal to a given number
- BIN-TREE – optimal binary decision tree (the tree that identifies all objects with a minimal amount of tests)
- SUBGRAPH-ISOMPOPHISM (but not GRAPH-ISOMPORPHISM)

revision

Performance ratios

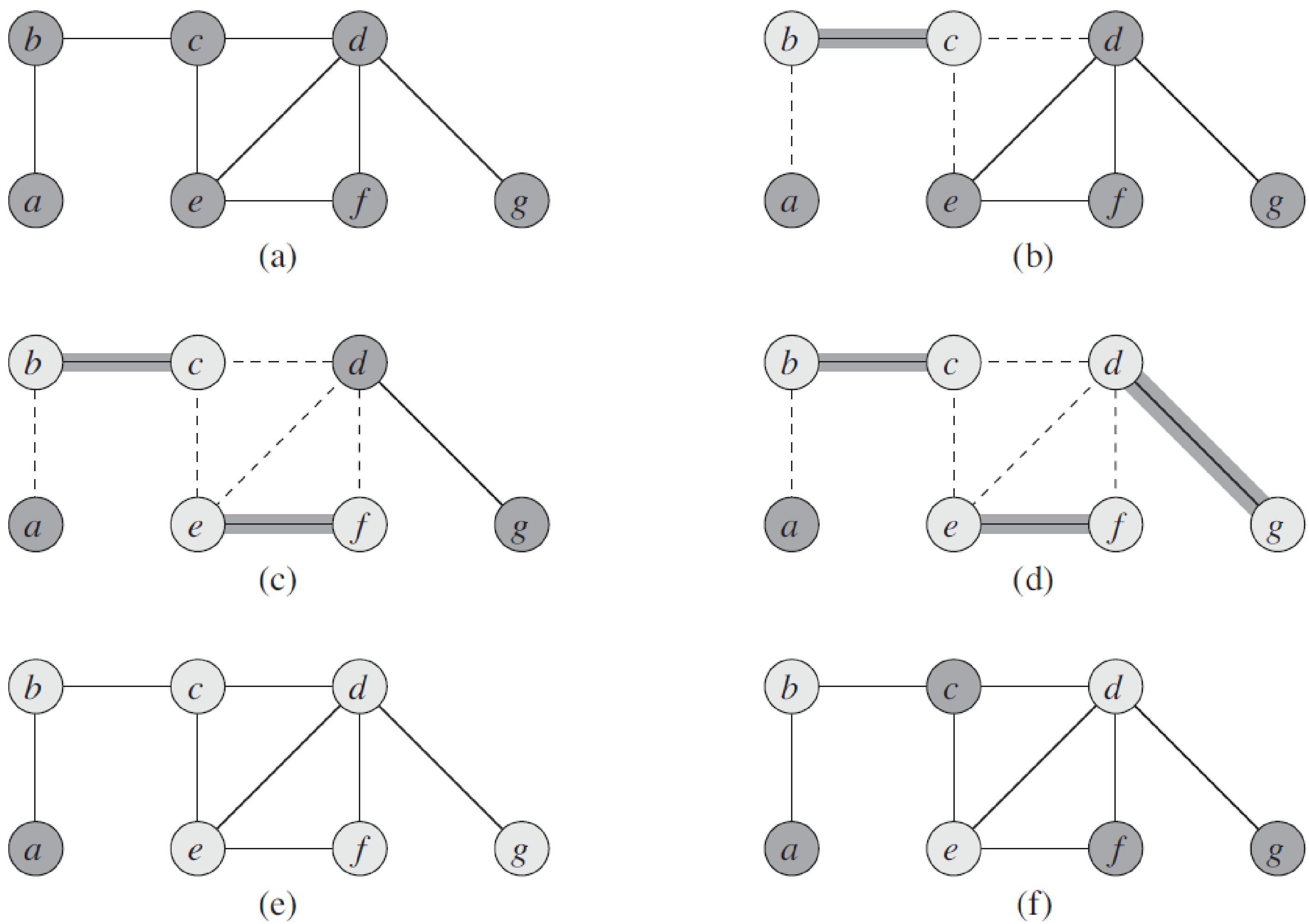
- approximation ratio is a ratio between the cost of approximate and optimal solution of a problem

- see Cormen et al: Introduction to algorithms, 2009/2022, Chapter 35

Vertex-cover

```
APPROX-VERTEX-COVER(G)
C = ∅
E' = G.E
while E' ≠ ∅
    let (u, v) be an arbitrary edge of E'
    C = C ∪ {u, v}
    remove from E' every edge incident on either u or v
return C
```

Illustration of Approx-Vertex-Cover algorithm



General TSP

- Non-existence of approximation algorithm for general TSP

MAX-3CNF-SAT

- expected approximation ratio is an expected ratio between the cost of approximate and optimal solution of a problem of a randomized algorithm
- randomized algorithm: randomly assign each of the variables with 0 or 1 with probability 0.5
- this algorithm is 8/7-approximation algorithm