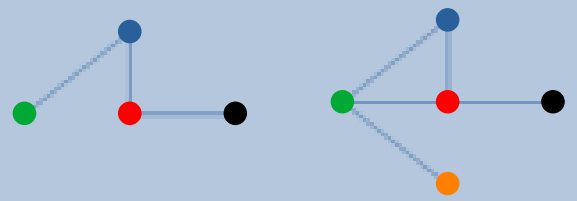
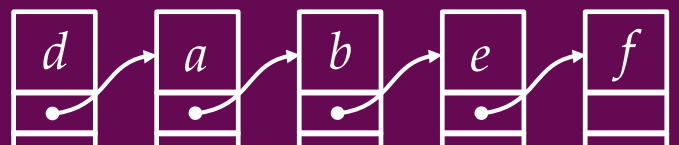
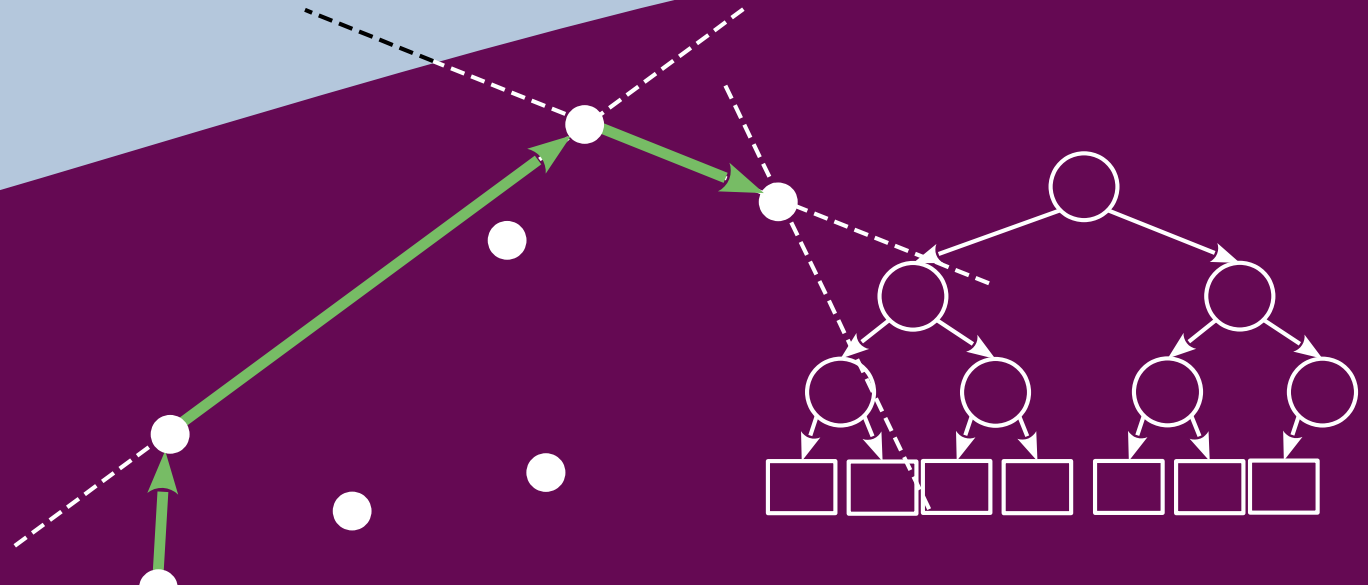
Advanced algorithms and structures data

12th lecture: Approximate algorithms

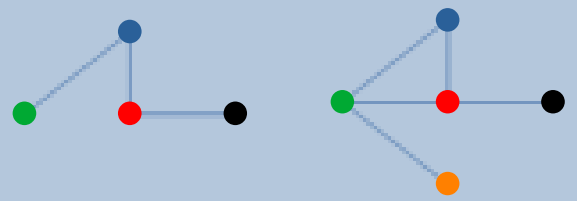


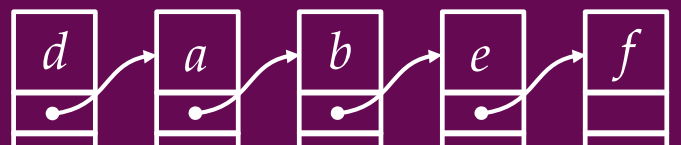


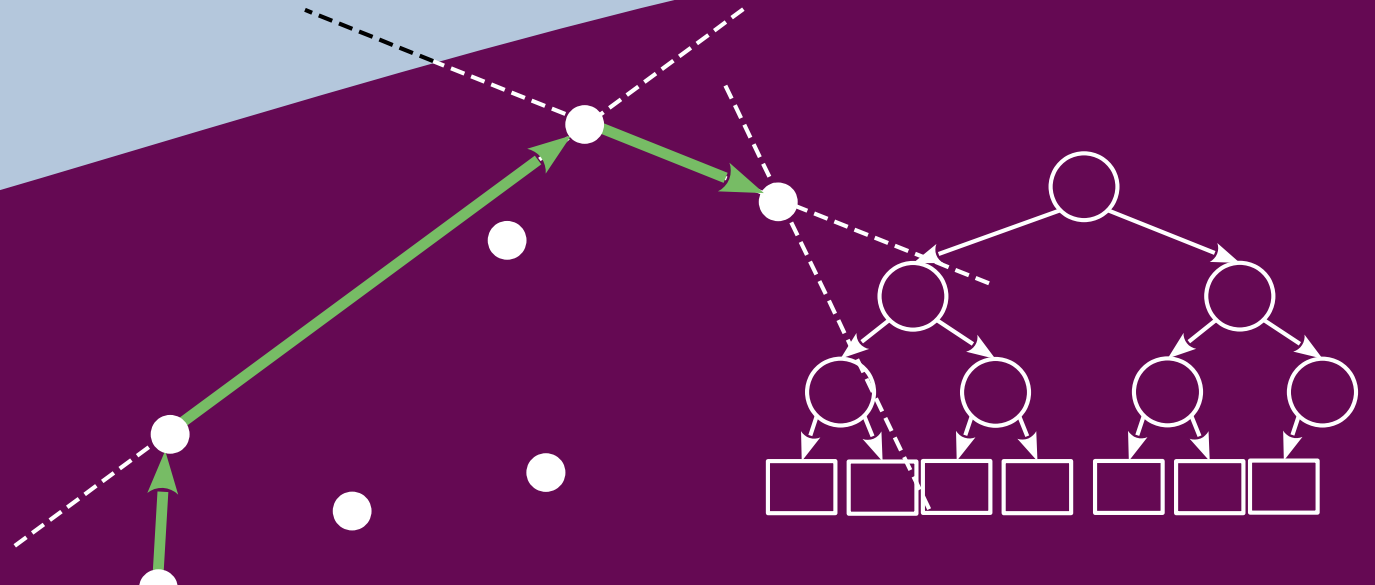


Napredni algoritmi i strukture podataka

12. predavanje: Približni algoritmi







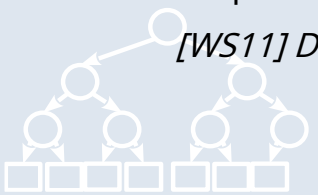
Approximate algorithms

- The basics
- MTSP – 2-approximation algorithm
- Vertex Cover – 2-approximation algorithm
- 0-1 knapsack – FPTAS

Lecture based on:

Script "Advanced algorithms and data structures", 2022.

[WS11] DP Williamson, D. Shmoys, "The Design of Approximation Algorithms", 2011, **subchapters 1.1.-1.3, 2.4, 3.1**



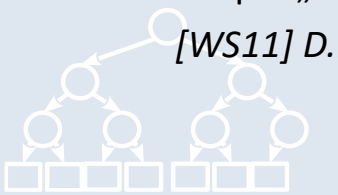
Približni algoritmi

- Osnove
- MTSP – 2-približni algoritam
- Vertex Cover – 2-približni algoritam
- 0-1 knapsack – FPTAS

Predavanje bazirano na :

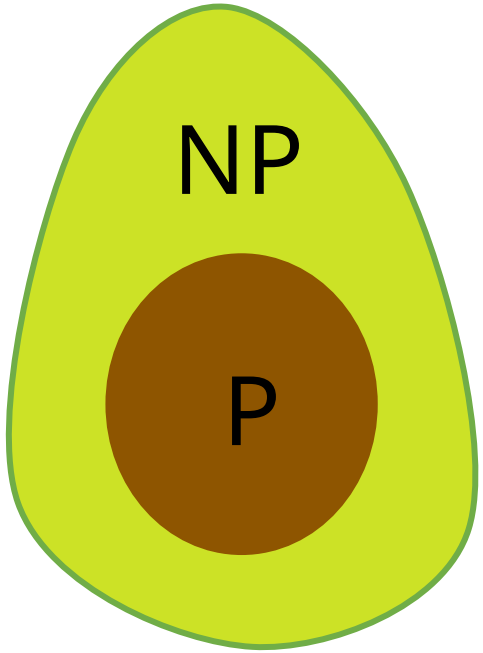
Skripta „Advanced algorithms and data structures”, 2022.

[WS11] D.P. Williamson, D. Shmoys „„The Design of Approximation Algorithms”, 2011; **potpoglavlja 1.1.-1.3, 2.4, 3.1**



Approximate algorithms?

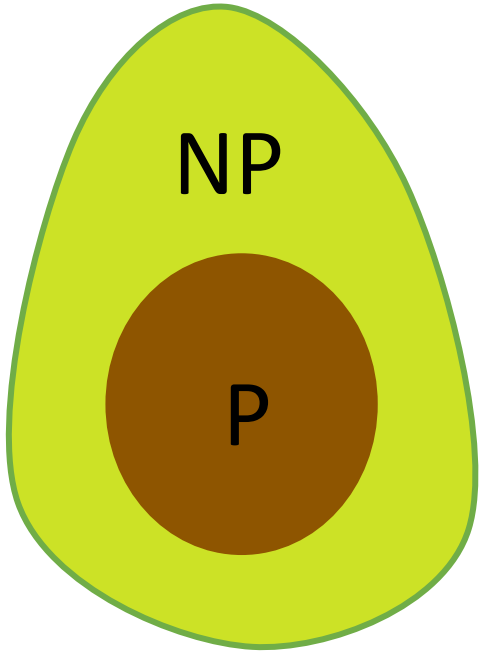
- =?



- NP-hard discrete problems
  - We want to get the best possible solution in polynomial time
  - Guarantees loss of performance
  - Resource-quality tradeoff
- APX class (approximate)

Približni algoritmi?

- $NP =^{?} P$



- NP-teški diskretni problemi
  - Želimo dobiti što bolje rješenje u polinomijalnom vremenu
  - Garancije gubitka performanse
  - Tradeoff resurs-kvaliteta

- APX klasa (aproksimabilni)

Approximate algorithms?

- Common tools for the design of approximate algorithms
  - Greedy algorithms
  - Local search
  - Dynamic programming
  - Randomization
  - Quantization (rounding)
    - Basic
    - Adaptive
    - By accident
  - Convex optimization (eg LP)

Približni algoritmi?

- Česti alati za dizajn približnih algoritama
  - Pohlepni algoritmi
  - Lokalno pretraživanje
  - Dinamičko programiranje
  - Randomizacija
  - Kvantizacija (zaokruživanje)
    - Osnovno
    - Adaptivno
    - Slučajno
  - Konveksna optimizacija (npr. LP)

α–approximate algorithm

- α–approximate optimization algorithm
  - Time polynomial
  - Solution zu**the worst**case within**factor α**from the optimum x*
    - To maximizez ≥ ⋅ *, < 1
    - To minimizez ≤ ⋅ *, > 1

α–približni algoritam

- α–približni algoritam za optimizaciju
  - Vremenski polinomijalan
  - Rješenje z u **najgorem** slučaju unutar **faktora α** od optimuma x*
    - Za maksimizaciju $z \geq \alpha \cdot x^*, \alpha < 1$
    - Za minimizaciju $z \leq \alpha \cdot x^*, \alpha > 1$

PTAS

- Time polynomial approximation scheme
  - English polynomial-time approximation scheme (PTAS)
  - Family of algorithms $\{A_\varepsilon\}$ for every $\varepsilon > 0$ there is a A_ε such that:
    - To maximize $(1-\varepsilon)$-approximate algorithm 1
    - To minimize $(1+\varepsilon)$-approximate algorithm

- A recipe, a meta-algorithm for the construction of approximate algorithms
  - Parameter

PTAS

- Vremenski polinomijalna približna shema
  - engl. polynomial-time approximation scheme (PTAS)
  - Familija algoritama$\{A_\epsilon\}$, za svaki $\epsilon > 0$ postoji A_ϵ takav da je:
    - Za maksimizaciju $(1 - \epsilon)$- približni algoritam
    - Za minimizaciju $(1 + \epsilon)$- približni algoritam

- Recept, meta-algoritam za konstrukciju približni algoritama
  - Parametar ϵ

PTAS

- A recipe, a meta-algorithm for the construction of approximate algorithms
  - Parameter

- Polynomial with respect to the input problem,**not necessarily** with 1/

PTAS

- Recept, meta-algoritam za konstrukciju približni algoritama
  - Parametar ϵ

- Polinomijalan sa obzirom na ulazni problem, **ne nužno** s $1/\epsilon$

FPTAS

- Even more restrictive!

- Time fully polynomial approximation scheme
  - English fully polynomial-time approximation scheme (FPTAS)
  - PTAS such that the execution time of each runtime bounded from above by the polynomial u1/

FPTAS

- Još restriktivnije!

- Vremenski potpuno polinomijalna približna shema
  - engl. fully polynomial-time approximation scheme (FPTAS)
  - PTAS takav da je vrijeme izvođenja svakog A_ϵ vrijeme izvođenja ograničeno odozgo polinomom u $1/\epsilon$

Approximate algorithms

- Three key questions for every candidate

1. Correctness - a feasible solution?
2. Efficiency – polynomial time?
3. Quality - strict guarantees on the distance from the optimum?

Približni algoritmi

- Tri ključna pitanja za svakog kandidata

1. Ispravnost – izvedivo rješenje?
2. Efikasnost – polinomijalno vrijeme?
3. Kvaliteta – striktne garancije na udaljenost od optimuma?

Examples of INAPPROPRIATE problems

- Non-approximable in polynomial time (unless P=NP)
  - The general problem of the traveling salesman
  - Maximum click
  - Maximum independent set

Primjeri NEprikladnih problema

- Neaproksimabilni u polinomijalnom vremenu (osim ako P=NP)
  - Općeniti problem trgovačkog putnika
  - Maksimalna klika
  - Maksimalni nezavisni skup

Examples of suitable problems

- The traveling salesman's metric problem
  - minimization
- The vertex cover problem
  - minimization
- 0-1 knapsack problem
  - maximization

Primjeri prikladnih problema

- Metrički problem trgovačkog putnika
  - minimizacija
- Problem vršnog pokrivača (vertex cover)
  - minimizacija
- 0-1 problem naprtnjače
  - maksimizacija

Metric Salesman Problem (MTSP)

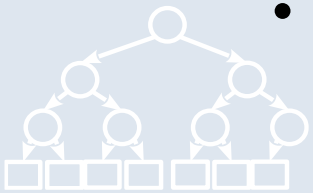
- TSP + triangle inequality in distances
- NP-hard problem

- 2-approximation (2-MST heuristic)
  - Naive Eulerization of MST
- 3/2-approximation (Christofides, 1976)
  - Eulerization of MST à la CPP
- (3/2 −)approximation (Karlin et al., 2020)
  - Christofides,**random tree**instead of MST

Metrički problem trgovačkog putnika (MTSP)

- TSP + nejednakost trokuta u udaljenostima
- NP-težak problem

- 2-aproksimacija (2-MST heuristika)
  - Naivna Eulerizacija MST-a
- 3/2-aproksimacija (Christofides, 1976)
  - Eulerizacija MST-a à la CPP
- $(3/2 - \epsilon)$ aproksimacija (Karlin et al., 2020)
  - Christofides, **slučajno stablo** umjesto MST



Metric Salesman Problem (MTSP)

- NP-hard problem

- The limit of approximability?
  - NP-hard to approximate with a factor < 123/122 (Karpinski et al. in 2013)

Metrički problem trgovačkog putnika (MTSP)

- NP-težak problem

- Granica aproksimabilnosti?
  - NP-teško aproksimirati sa faktorom $\alpha < 123/122$ (Karpinski et al. 2013)

2-MST heuristics

Entrance: (, ×)

1. Find the MST in G, the weight***x***
   $O(|V|_2)$
2. DFS tour of all edges twice, write vertices in sheet L, length of tour***2x*** $O(|V|)$

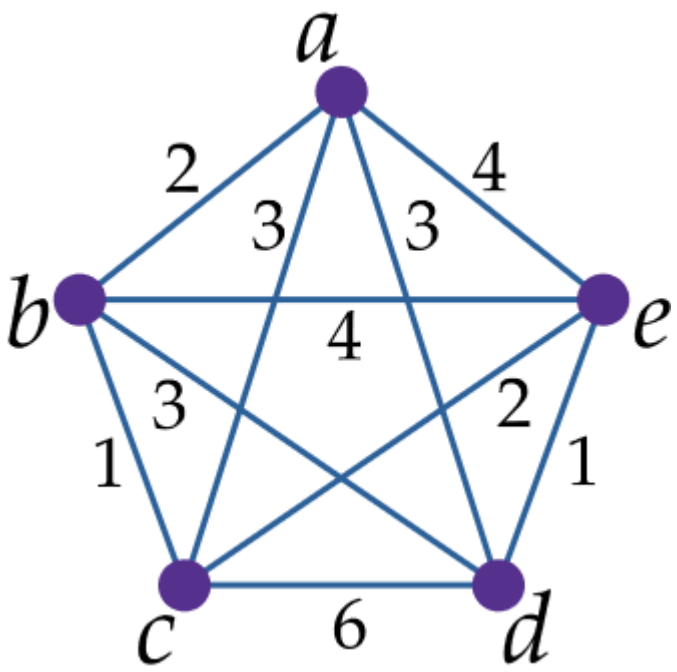
3. Filter L keeping only the first occurrences of vertices (short connection) and store in FL, the length of the tour***z*** $O(|V|)$

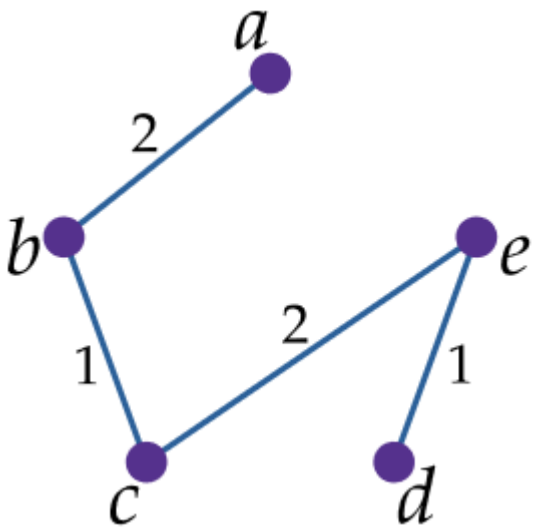
2-MST heuristika

Ulaz: $G(V, V \times V)$

1. Pronađi MST u G, težina ***x***
   $O(|V|^2)$
2. DFS obilazak svim bridovima dvaput, zapisati vrhove u listu L, duljina obilaska ***2x***
   $O(|V|)$
3. Filtriraj L čuvajući samo prva pojavljivanja vrhova (kratko spajanje) i spremi u FL, duljina obilaska ***z***
   $O(|V|)$

2-MST heuristics - an example





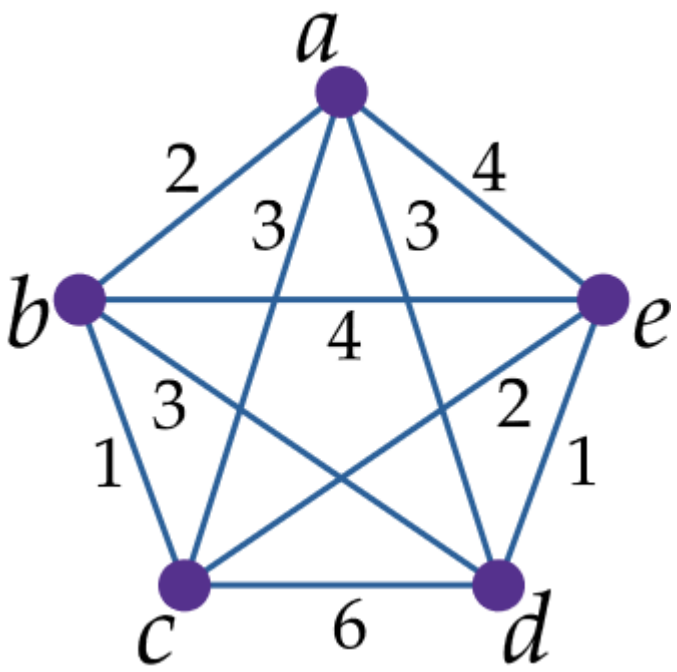
1.MST

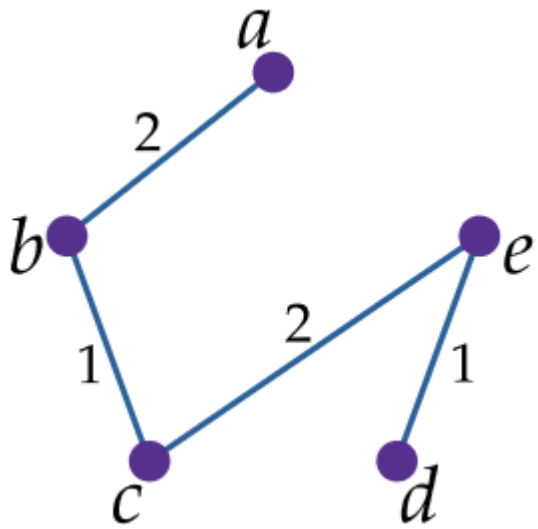
2. From the top a

L=[a,b,c,e,d,e,c,b,a]

3.*FL = [a,b,c,e,d]*

2-MST heuristika - primjer





1.MST

2. Iz vrha a

L=[a,b,c,e,d,e,c,b,a]

3. *FL =[a,b,c,e,d]*

2-MST heuristics: analysis

- **Correctness**–FL contains each vertex once, and we interpret the ends as connected. A valid tour✓
- **Efficiency**–2-MST runs in polynomial time. ✓
- **Quality?**

2-MST heuristika: analiza

- **Ispravnost** – FL sadrži svaki vrh jednom, a krajeve interpretiramo kao spojene. Valjan obilazak ✓
- **Efikasnost** – 2-MST se izvodi u polinomijalnom vremenu. ✓
- **Kvaliteta?**

2-MST heuristics: quality

- **Lemma 12.3.** For the weight x of the MST, $x \leq z$ holds.

- **Lemma 12.4.** 2-MST is a 2-approximation algorithm. ✓

$$\leq \quad \leq 2$$

MST

Inequality triangle

2-MST heuristika: kvaliteta

- **Lema 12.3.** Za težinu x MST-a, vrijedi x≤z.

- **Lema 12.4.** 2-MST je 2-približni algoritam. ✓

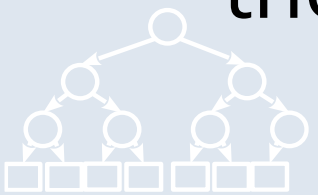
$$x \leq z \leq 2x$$

MST

Nejednakost trokuta

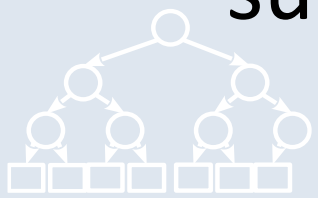
The peak cover problem

- Given an undirected graph (,)and peak costs : → ℝ
- **Top cover**is ′ ⊆ such that for every edge in E one of the vertices in '
- The top cover of the minimum cost – 'such that the sum of costs in it is minimal



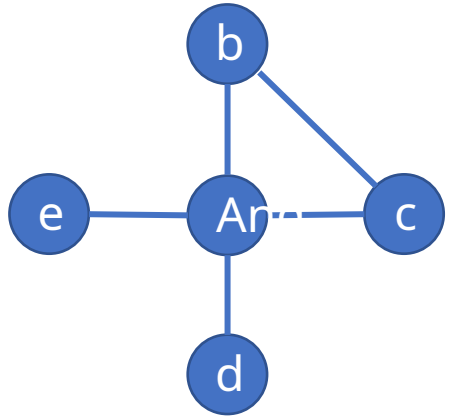
Problem vršnog pokrivača

- Dani neusmjereni graf $G(V, E)$ i vršni troškovi $c: V \rightarrow \mathbb{R}$
- **Vršni pokrivač** je $V' \subseteq V$ takav je za svaki brid u E jedan od vrhova unutar V’
- Vršni pokrivač minimalnog troška – V’ takav da je suma troškova u njemu minimalna



The peak cover problem

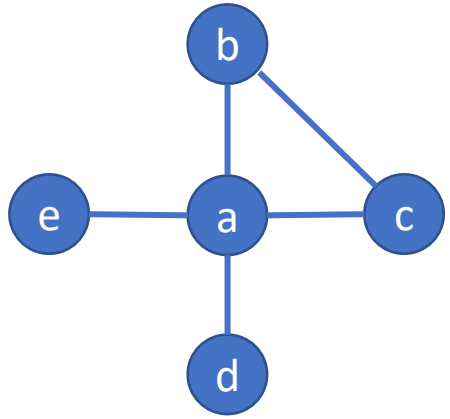
- The top cover of the minimum cost – 'such that the sum of costs in it is minimal



- A simple 2-approximation algorithm based on:
  - linear programming and
  - deterministic rounding of decimal numbers

Problem vršnog pokrivača

- Vršni pokrivač minimalnog troška – V' takav da je suma troškova u njemu minimalna



- Jednostavni 2-približni algoritam baziran na:
  - linearnom programiranju i
  - deterministčkom zaokruživanju decimalnih brojeva

The peak cover problem

- Modeled as an integer linear program

$$\min$$

$$\geq 1$$

$$\in 0.1\{\quad\}| \ |$$

-incidence matrix

$$= o0 \quad \notin \prime$$

$$1 \quad \in \prime$$

- NP-hard, optimal *

Problem vršnog pokrivača

- Modeliran kao cjelobrojni linearni program

$$\min c^T x$$
$$Ax \geq 1$$
$$x \in \{0,1\}^{|V|}$$

A - matrica incidencije

$$x_i = \begin{cases} 0 \; ako \; i \notin V' \\ 1 \; ako \; i \in V' \end{cases}$$

- NP-teško, optimum x^*_{ILP}

The peak cover problem - relaxation

- Continuous range

$$\min$$
$$\geq 1$$
$$\in [0,1] \quad | \ |$$

Efficient solution!

- Optimum $*$, potentially decimal

$$* \leq \quad *$$

Problem vršnog pokrivača - relaksacija

- Kontinuirani raspon

$$\min c^T x$$
$$Ax \geq 1$$
$$x \in [0,1]^{|V|}$$

Efikasno rješavanje!

- Optimum x^*, potencijalno decimalan

$$c^T x^* \leq c^T x^*_{ILP}$$

Approximate algorithm -DetRoundLP

Entrance: = (,), : → ℝ

1. ∗= LP relaxation solution

2. = (∗)

3. back ′= { ∈ | = 1}

Približni algoritam -DetRoundLP

Ulaz: $G = (V, E), c: V \to \mathbb{R}$

1. x^*= rješenje LP relaksacije

2. $z = round(x^*)$

3. vrati $V' = \{i \in V | z_i = 1\}$

DetRoundLP: analysis

- **Correctness**–the sum of two variables on the edges at least
  1. There must be at least one≥ 0.5
     - Rounding selects at least one incident vertex for each edge. Rolled top cover✓

- **Efficiency**–each step can be done in polynomial time (and solving LP)✓

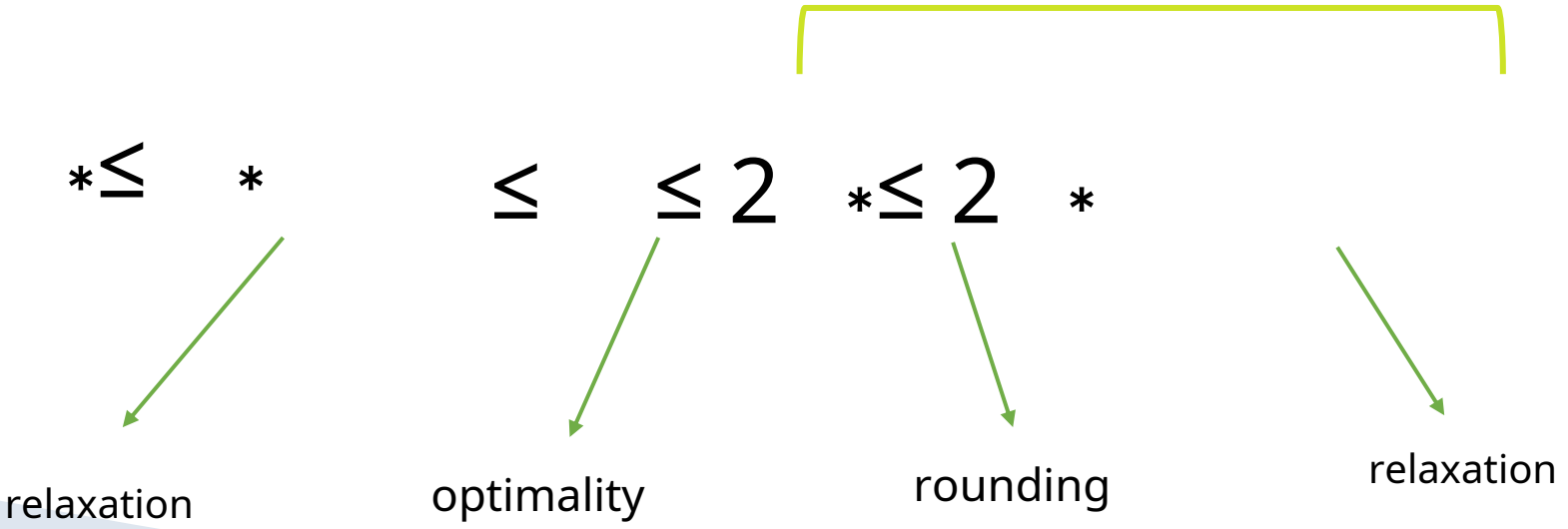
DetRoundLP: analiza

- **Ispravnost** – sume dvije varijable na bridovima barem 1. Barem jedna mora biti ≥ 0.5
  - Zaokruživanje za svaki brid odabere barem jedan incidentni vrh. Valjani vršni pokrivač ✓

- **Efikasnost** – svaki korak se može obaviti u polinomijalnom vremenu (i rješavanje LP) ✓

DetRoundLP: quality

- **Lemma 12.2.** DetRoundLP is a 2-approximation algorithm.



DetRoundLP: kvaliteta

- **Lema 12.2.** DetRoundLP je 2-približni algoritam.

$$c^T x^* \leq c^T x^*_{ILP} \leq c^T z \leq 2c^T x^* \leq 2c^T x^*_{ILP}$$

relaksacija

optimalnost

zaokruživanje

relaksacija

DetRoundLP: quality

- **Lemma 12.2.** DetRoundLP is a 2-approximation algorithm.

- A strict upper limit?
  - There are instances of graphs for which DetRoundLP produces a solution twice worse than the optimum

DetRoundLP: kvaliteta

- **Lema 12.2.** DetRoundLP je 2-približni algoritam.

- Striktna gornja granica?
  - Postoje instance grafova za koje DetRoundLP proizvodi rješenje dvostruko gore od optimuma

Vertex cover – limits of approximability

- [WS11] Ak o there is an α-approximate algorithm with $<$ $-$ $\approx$. ,then P=NP

- If the unique games conjecture is true, the upper bound becomes

$<$

Vršni pokrivač – granice aproksimabilnosti

- [WS11] Ako postoji α-približni algoritam sa $\boldsymbol{\alpha < 10\sqrt{5} - 21 \approx 1.36}$, onda P=NP

- Ako je istinita konjektura jedinstvenih igara (engl. unique games conjecture), gornja granica postaje $\boldsymbol{\alpha < 2}$

0-1 knapsack

- **NP-hard problem**
- Things $= \{1, \ldots, \}$, each size $\in \mathbb{N}$, and values $\in \mathbb{N}$
- Backpack capacity $\in \mathbb{N}$
- Find the subset of things that fit in the knapsack and have the maximum sum of values

0-1 knapsack

- **NP-težak problem**

- Stvari $I = \{1, \dots, n\}$, svaka veličine $s_i \in \mathbb{N}$, i vrijednosti $v_i \in \mathbb{N}$
- Kapacitet naprtnjače $B \in \mathbb{N}$
- Pronaći podskup stvari koje stanu u naprtnjaču i imaju maksimalnu sumu vrijednosti

0-1 Knapsack

- **NP-hard problem**
- Dealing with DP –**pseudo-polynomial algorithm**
  - ( ) - numeric parameter
- Unary encoding – encoding with consecutive units
- Def. It's an algorithm**pseudo-polynomial**if performed in time to a polynomial input when the numeric part is encoded**unary**(rather than binary).

0-1 Knapsack

- **NP-težak problem**
- Rješavanje s DP – **pseudo-polinomijalni algoritam**
  - $O(nB)$ - B numerički parametar
- Unarno enkodiranje – enkodiranje uzastopnim jedinicama
- Def. Algoritam je **pseudo-polinomijalan** ako se izvodi u vremenu polinomijalnom ulazu kada je numerički dio enkodiran **unarno** (a ne binarno).

0-1 Knapsack

- If the numerical parameters are polynomial in ***n***
  - Polynomial algorithm!!

- FPTAS – aggregation of numerical parameters into compartments
  - The number of compartments depends polynomially on ***n***

0-1 Knapsack

- Ako su numerički parametri polinomijalni u ***n***
  - Polinomijalan algoritam!!

- FPTAS – agregiranje numeričkih parametara u pretince
  - Broj pretinaca ovisi polinomijalno o ***n***

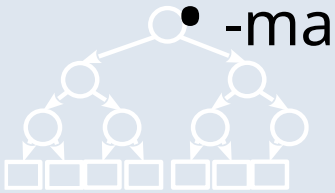
0-1 Knapsack – new DP table!

- Table - columns of things, rows of knapsack values
- Cell [,] ->the least necessary cost that realizes value using some of the first things.

- Change required for approximate algorithm
  - Check what happens when you try the same trick with the following slides over a regular (costs, things) table

- The value of the most valuable thing = max

- -max value that can be achieved by a knapsack with n things and N

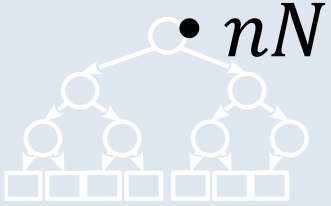


0-1 Knapsack – nova DP tablica!

- Tablica – stupci stvari, retci vrijednosti naprtnjače
- Ćelija $A[v, i]$ -> najmanji potrebni trošak koji ostvaruje vrijednost v koristeći neke od prvih i stvari.

- Potrebna promjena za približni algoritam
  - Provjerite što se događa kad probate isti trik sa narednih slideova nad uobičajenom (troškovi, stvari) tablicom

- Vrijednost najvrjednije stvari $N = \max_i v_i$
- nN - max vrijednost koju može ostvariti knapsack sa n stvari i N



0-1 Knapsack – DP algorithm!

DPKnapsackByValue

1. [: , 1] = + 1
2. 0,1[= 0] 1, 1 =[min(1, 1 , 1) [ ]
3. h = 2, . . ,
   1. [: ,] = [: , − 1]
   2. h = , … ,
      1. [,] = min([, − 1], [− , − 1] +)
4. max{ : , ≤ } [ ]

2( )

pseudo-

polynomial on

- works by N

0-1 Knapsack – DP algoritam!

DPKnapsackByValue

1. $A[:,1] = B + 1$
2. $A[0,1] = 0, A[v_1, 1] = \min(A[v_1, 1], s_1)$
3. $For\ each\ i = 2,..,n$
   1. $A[:,i] = A[:,i-1]$
   2. $For\ each\ v = v_i, \ldots.., nN$
      1. $A[v,i] = \min(A[v,i-1], A[v - v_i, i-1] + s_i)$
4. $Return\ \max\{v: A[v,n] \leq\ B\}$

$O(n^2N)$ pseudo-polinomijalno - radi N

0-1 Knapsack – approximation

- We must limit N by a polynomial in n
- Quantization of values to units

BucketizedDP for knapsack – parameter

1. =
2. ′ = ,

3. Fix modified problem using DPKnapsackByValue

0-1 Knapsack – aproksimacija

- Moramo ograničiti N polinomom po n
- Kvantizacija vrijednosti na jedinice μ

BucketizedDP za knapsack – parametar ϵ

1. $\mu = \frac{\epsilon N}{n}$
2. $v_i' = \left\lfloor \frac{v_i}{\mu} \right\rfloor, \forall i \in I$
3. Riješi izmijenjeni problem koristeći DPKnapsackByValue

Example 0-1 knapsack FPTAS

- Solve the 8 capacity backpack problem with the following 5 things **0.25-approximate algorithm**

	1	2	3	4	5
c	2	4	8	16	20
with	1	4	2	5	7

Primjer 0-1 knapsack FPTAS

- Riješite problem naprtnjače kapaciteta 8 sa sljedećih 5 stvari **0.25-približnim algoritmom**

	1	2	3	4	5
v	2	4	8	16	20
s	1	4	2	5	7

Example 0-1 knapsack FPTAS

- 0.25-approximate algorithm

	1	2	3	4	5
c	2	4	8	16	20
with	1	4	2	5	7

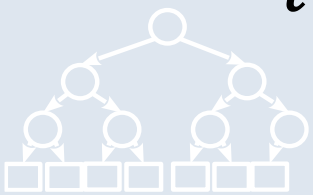
- = 8, = 20
- = 0.75
- = = 0.75 · 2 0 = 3 (5)
- $v'_i = \left\lfloor \frac{v_i}{K} \right\rfloor, \forall v_i \in V$ → v'=[0,1,2,5,6]

Primjer 0-1 knapsack FPTAS

- 0.25-približni algoritam

	1	2	3	4	5
v	2	4	8	16	20
s	1	4	2	5	7

- $B = 8, N = 20$
- $\epsilon = 0.75$
- $\mu = \frac{\epsilon N}{n} = \frac{0.75 \cdot 20}{5} = 3$
- $v_i' = \left\lfloor \frac{v_i}{\mu} \right\rfloor, \forall i \in I$ ⟶ v'=[0,1,2,5,6]



0-1 knapsack FPTAS: analysis

- **Correctness**–DP returns a solution that satisfies the capacity constraint✓

- **Efficiency**– is transformed into after scaling . Complexity of the interlinked DP is ( ) ✓

- **Quality?**

0-1 knapsack FPTAS: analiza

- **Ispravnost** – DP vraća rješenje koje zadovoljava ograničenje kapaciteta ✓

- **Efikasnost** – N se nakon skaliranja transformira u $\frac{n}{\epsilon}$. Složenost upretinčenog DP jest $O(\frac{n^3}{\epsilon})$ ✓

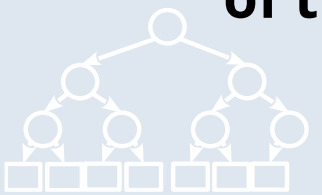
- **Kvaliteta?**

0-1 knapsack FPTAS: quality

- **Lemma 12.5.** The presented algorithm is FPTAS for 0-1 knapsack. ✓

- That is the solution is of value at least (1 −)from the optimal value

- Proof in script, based on **to the method of construction of the size of the compartments**



0-1 knapsack FPTAS: kvaliteta

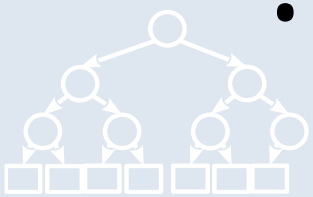
- **Lema 12.5.** Prezentirani algoritam jest FPTAS za 0-1 naprtnjaču. ✓

- Tj. rješenje je vrijednosti barem $(1 - \epsilon)$ od optimalne vrijednosti

- Dokaz u skripti, baziran na **načinu konstrukcije veličine pretinaca**

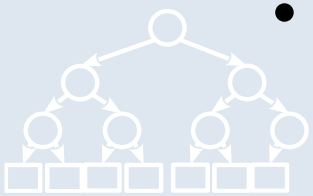
Conclusion

- For some problems we can find approximate algorithms
  - ***MTSP – 2-approximate, 1.5-approximate***
    - ***NP-hard for*** $<$ ——
  - ***Peak cover – 2-approximate***
    - ***NP-hard for*** $<$. , ***MAYBE even*** $<$
  - ***0-1 knapsack – FPTAS***
    - ***A family of algorithms for everyone*** $\in(,)$
- For some problems we can't handle any
  - TSP, maximal clique and maximal anticlique



Zaključak

- Za neke probleme možemo naći približne algoritme
  - ***MTSP – 2-približni, 1.5-približni***
    - ***NP-teško za*** $\alpha < \frac{123}{122}$
  - ***Vršni pokrivač – 2-približni***
    - ***NP-teško za*** $\alpha < 1.36$***, MOŽDA čak i*** $\alpha < 2$
  - ***0-1 knapsack – FPTAS***
    - ***Familija algoritama za sve*** $\alpha \in (0, 1)$
- Za neke probleme NE možemo ni za koji α
  - TSP, maksimalna klika i maksimalna antiklika



Conclusion

- We used the following techniques in the design of the algorithms
  - Quantization (rounding)
    - Adaptive input rounding - knapsack
    - Basic output rounding – top cover
  - Linear programming – vertex cover
  - Dynamic programming - knapsack
  - Greedy algorithms – MTSP
  - Local search - MTSP

Zaključak

- Koristili smo sljedeće tehnike u dizajnu algoritama
  - Kvantizacija (zaokruživanje)
    - Adaptivno zaokruživanje ulaza – knapsack
    - Osnovno zaokruživanje izlaza – vršni pokrivač
  - Linearno programiranje – vršni pokrivač
  - Dinamičko programiranje – knapsack
  - Pohlepni algoritmi – MTSP
  - Lokalno pretraživanje - MTSP