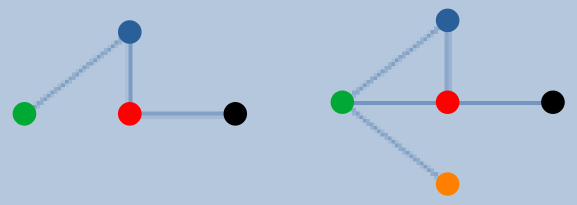
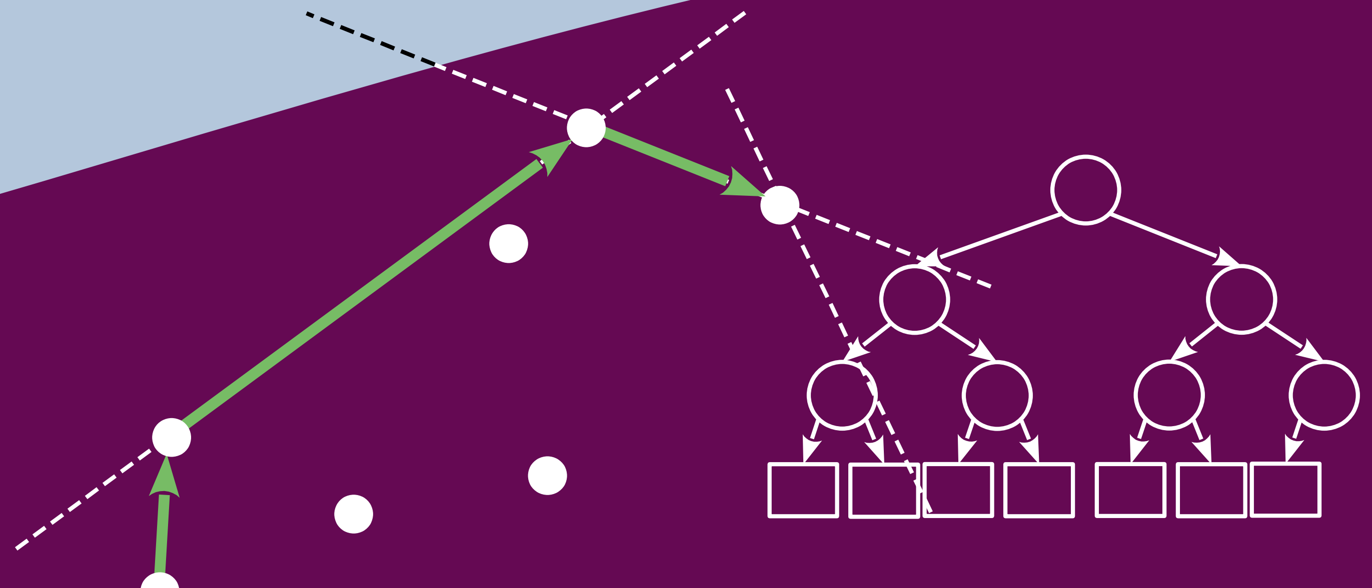
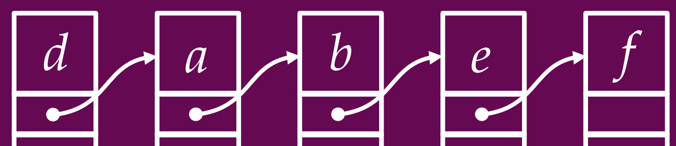
Napredni algoritmi i strukture podataka

12. predavanje: Približni algoritmi







Nasumični algoritmi

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

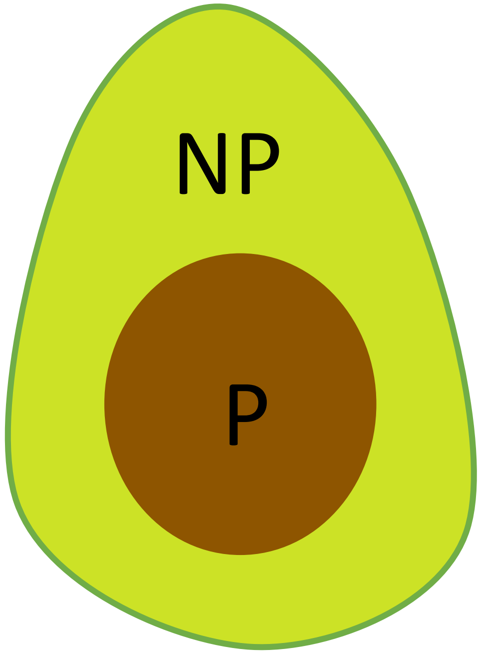
Predavanje bazirano na :

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**

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)



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–približni algoritam

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

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

- Recept, meta-algoritam za konstrukciju približni algoritama
  - Parametar ϵ
- Polinomijalan sa obzirom na ulazni problem, **ne nužno** s $1/\epsilon$

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 $1/\epsilon$

Približni algoritmi

- Tri ključna pitanja za svakog kandidata

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

Primjeri NEprikladnih problema

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

Primjeri prikladnih problema

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

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

Metrički problem trgovačkog putnika (MTSP)

- NP-težak problem

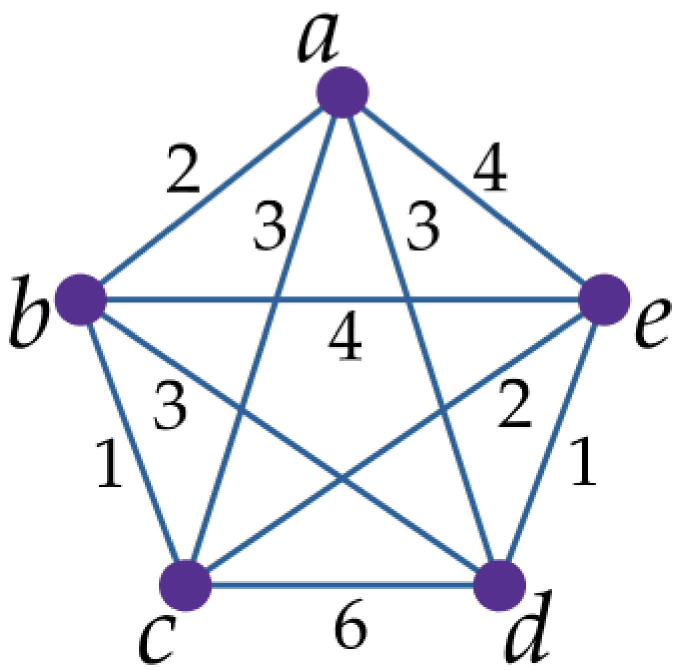
- Granica aproksimabilnosti?
  - NP-teško aproksimirati sa faktorom $\alpha < 123/122$ (Karpinski et al. 2013)

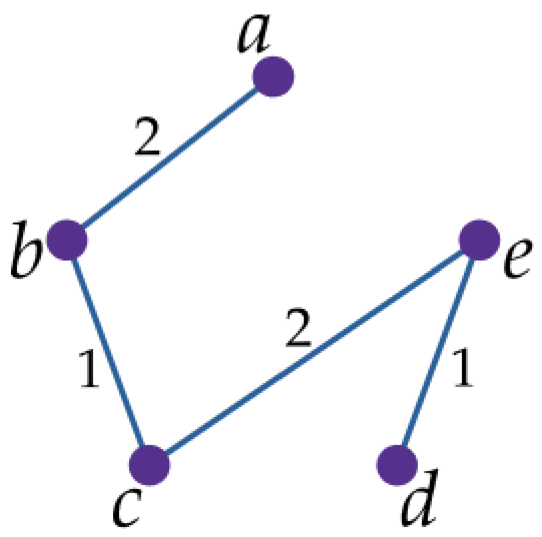
2-MST heuristika

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 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 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 heuristika: kvaliteta

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

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

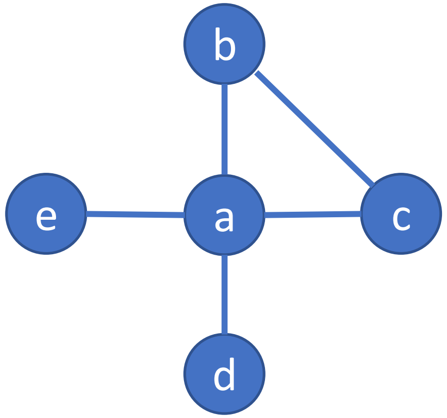
$$\underbrace{x \leq}_{\text{MST}} z \underbrace{\leq 2x}_{\text{Nejednakost trokuta}}$$

Problem vršnog pokrivača

- Dani neusmjereni graf $G(V, E)$ i vršne troškove $c: V \to \mathbb{R}$
- **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

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
  - determinističkom zaokruživanju decimalnih brojeva

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}

Problem vršnog pokrivača - relaksacija

- Kontinuirani raspon

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

Efikasno rješavanje!

- Optimum x^*, potencijalno decimalan

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

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: 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: kvaliteta

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

$$c^T x^* \underset{\text{relaksacija}}{\leq} c^T x^*_{ILP} \underset{\text{optimalnost}}{\leq} c^T z \underset{\text{zaokruživanje}}{\leq} 2c^T x^* \underset{\text{relaksacija}}{\leq} 2c^T x^*_{ILP}$$

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

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-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-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

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

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

0-1 Knapsack – nova DP tablica!

- Retci – stupci vrijednosti, retci vrijednosti naprtnjače
- Ćelija $A[i, v]$ -> 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 narendih slideova nad uobičajenom (stvari,troškovi) 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 algoritam!

DPKnapsackByValue

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

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

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

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

Primjer 0-1 knapsack FPTAS

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

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

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: 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**

Zaključak

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

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