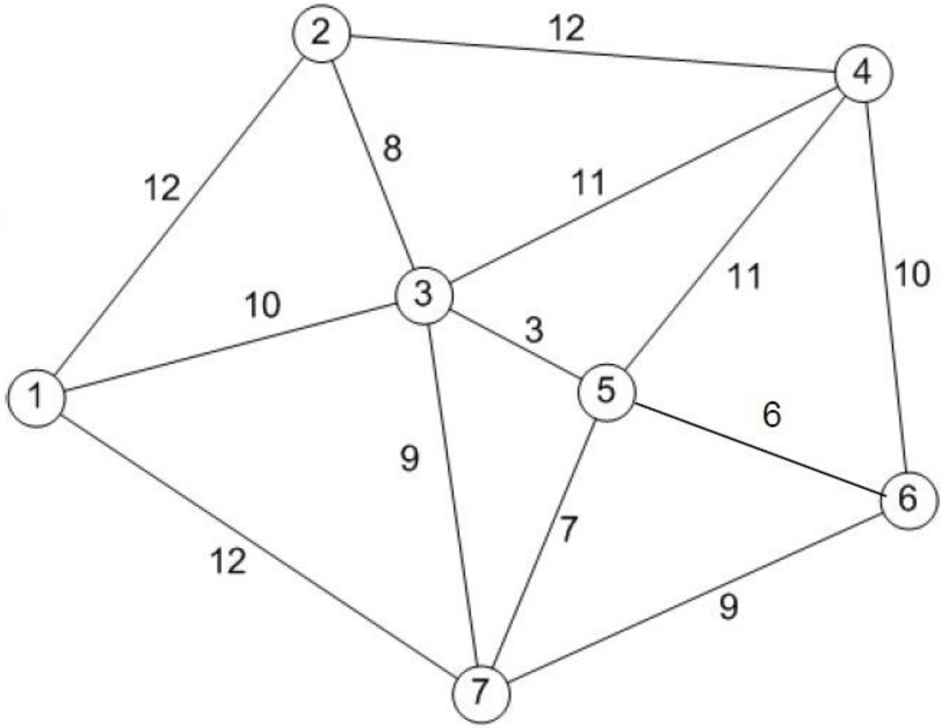
Dinamično programiranje

napredni pristopi

Priprave na računalniške olimpijade 2019/2020

Tomaž Hočevar

Potujoči trgovec (TSP)



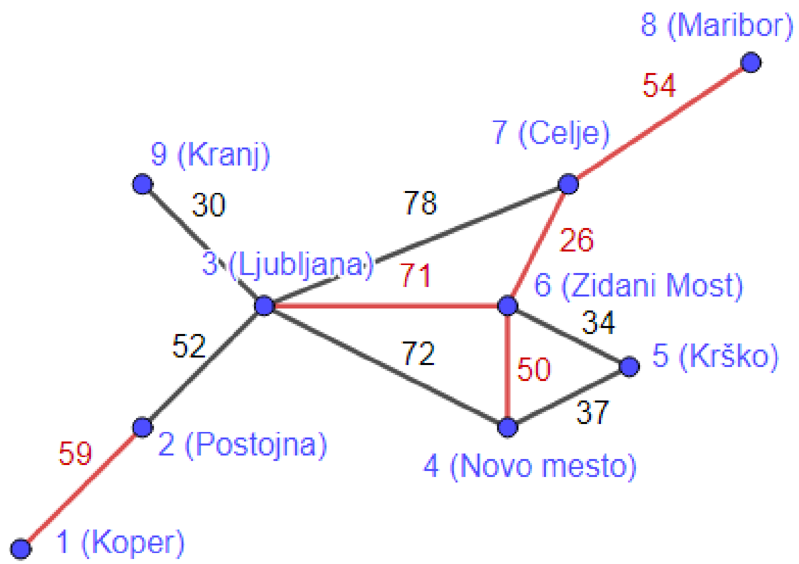
Potujoči trgovec (TSP)

- stanje/podproblem =

  (trenutno vozlišče, množica že obiskanih vozlišč)
- rešitev podproblema
  - preverimo vsa možna sosednja neobiskana vozlišča
- $O(n!) \dashrightarrow O(n^2 2^n)$
- implementacija
  - memoizacija rekurzivne rešitve
  - predstavitev množice z bitno masko

Steinerjevo drevo

- https://putka-upm.acm.si/tasks/t/avtocesta/



Koper – Postojna
Ljubljana – Maribor
Maribor – Novo mesto

Steinerjevo drevo

Steinerjevo drevo:

- najkrajše poti med vsemi pari točk – p(x,y)
- podproblem f(x,S) - Steinerjevo drevo s korenom v x, ki povezuje terminale S
  - odcep podmnožice terminalov T na vozlišču y
  - p(x,y) + f(y,T) + f(y,S-T)
- $O(3^n n^2)$

Avtocesta:

- preverimo vsa veljavna razbitja terminalov v komponente
- za vsako komponento imamo že izračunan rezultat

Najdaljše naraščajoče zaporedje (LIS)

x_i	0	8	4	12	2	10	6	14	1	9	5	13	3	11	7	15
f(i)																

Najdaljše naraščajoče zaporedje (LIS)

x_i	0	8	4	12	2	10	6	14	1	9	5	13	3	11	7	15
f(i)	1	2	2	3	2	3	3	4	2							

- $f(i) = \max\,(f(j)+1 \quad | \quad j < i \text{ in } x_j < x_i)$
- $O(n^2)$

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	2	2	0	2	0	3	0	2	0	3	0	3	0	4	0

- drevesna struktura
- $O(n \log n)$

Deli in vladaj (divide and conquer)

- https://putka-upm.acm.si/tasks/t/varianca/

> Razdeli zaporedje N števil na K strnjenih kosov tako, da minimiziraš varianco vsot po kosih.

								i		n
k=4	7	2	5	1	7	1	1	4	9	1
	7		v_2=8				13			10

$$avg = 1/K \sum_{i=1..K} v_i = 1/K \sum_{i=1..N} x_i$$

$$var = 1/K \sum_{i=1..K} (v_i - avg)^2$$

$$f(k, n) = \min_{i < n} f(k-1, i) + cost(i+1, n)$$

čas: $O(kn^2)$ ✗

Deli in vladaj (divide and conquer)

- monotonost delitvenih mest

  opt(k, n) = i, ki minimizira f(k, n)

  opt(k, n) <= opt(k, n+1)

k=4

7	2	5	1	7	1	1	4	9	1
7	2	5	1	7	1	1	4	9	
7	2	5	1	7	1	1	4		
7	2	5	1	7	1	1			

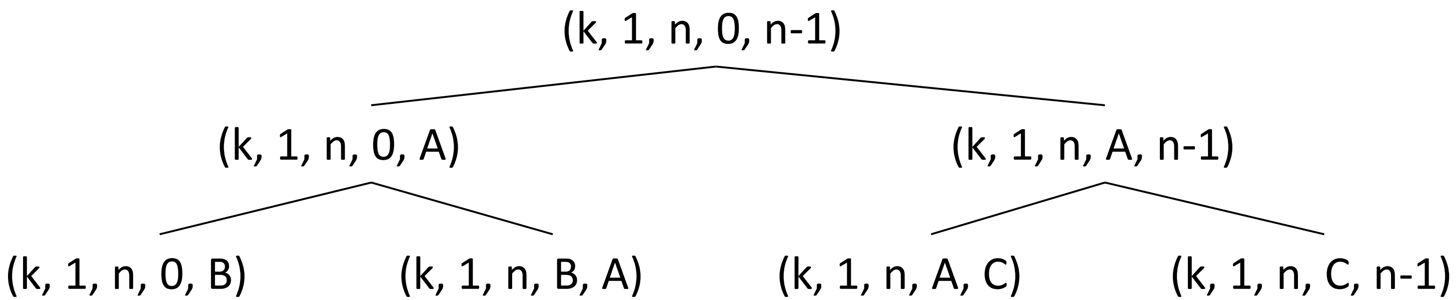
- optimizacija deli in vladaj
  - vrednosti f(k,n)
    - računamo naraščajoče po k
    - lahko v poljubnem vrstnem redu za n
  - O(k n^2) --> O(k n log n)

Deli in vladaj (divide and conquer)

solve(k, n_L, n_R, i_L, i_R)

- rešimo za $n = (n_L + n_R)/2$
- izračunamo f(k,n), opt(k, n)

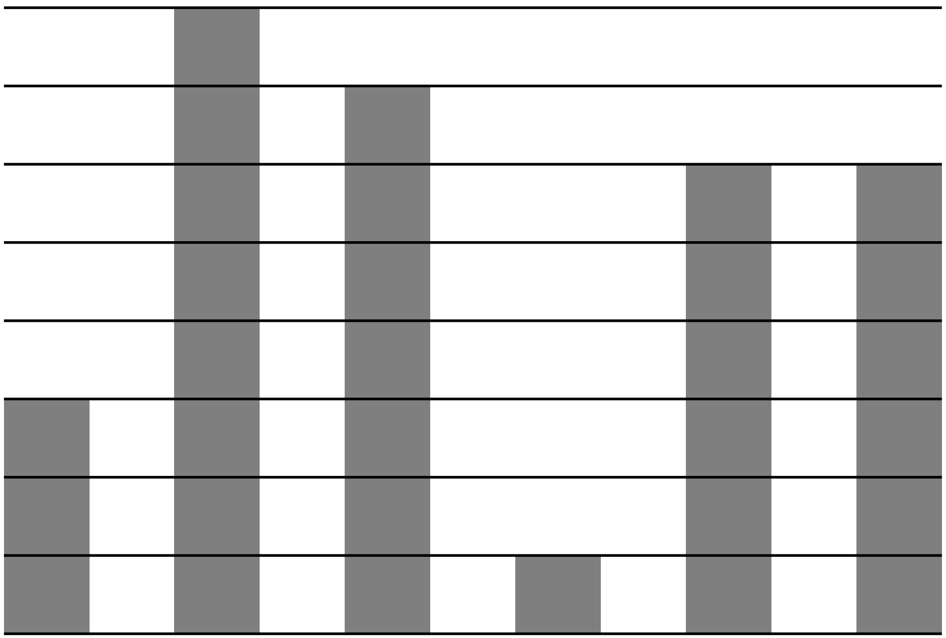
  $\min_{i_L <= i <= \min(i_R,\ n-1)} [\ f(k-1, n) + ((c(n)-c(i)) - avg)^2\ ]$
- solve(k, n_L, n-1, i_L, opt(k,n))
- solve(k, n+1, n_R, opt(k,n), i_R)



[0, n-1]
[0, A] + [A, n-1]
[0, B] + [B, A] + [A, C] + [C, n-1] <= 2n

Konveksna ovojnica (convex hull)

- http://putka-ceoi.fri.uni-lj.si/tasks/ceoi/day2/building



0		-1		9		1		2		0

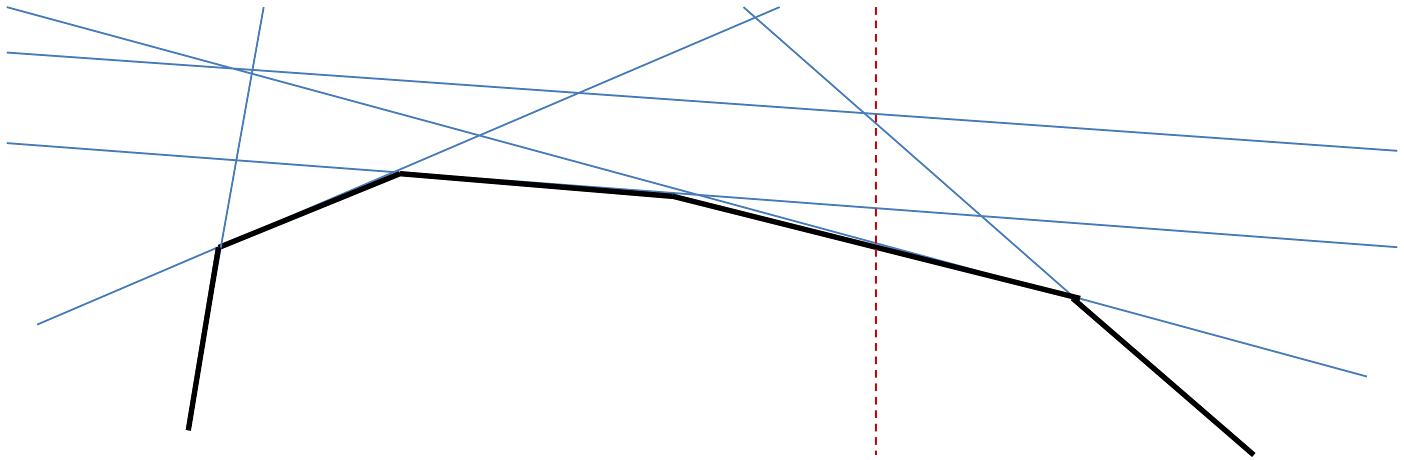
Konveksna ovojnica (convex hull)

$$f(i) = \min_{j<i} \left(f(j) + (h_j - h_i)^2 - w_i \right)$$

$$f(i) = \min_{j<i} \left(A(j) \cdot h_i + B(j) + C(i) \right)$$

$$A(j) = -2h_j \quad B(j) = h_j^2 + f(j) \quad C(i) = h_i^2 - w_i$$

Konveksna ovojnica (convex hull)



- daljice na ovojnici so urejene po naklonu in začetni x koordinati
- drevesna struktura daljic (x_i', k_i, n_i)
  - $y = k_i x + n_i$ za $x = [x_i', x_{i+1}']$
  - poizvedba: iskanje po x'
  - dodajanje: iskanje po k, brisanje nepomembnih