

# **Algoritmus brutální síly, greedy (hltaavý) algoritmus a inkrementální algoritmy**

---

I.Kolingerová

## 1. Brutální síla



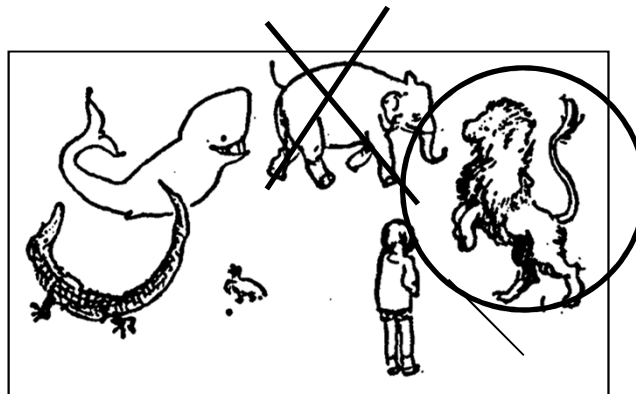
- Problém řešen prohledáním všech podmnožin, uspořádání nebo potencionálních řešení a výběrem nejlepšího
- Pro malé problémy postačující
- Často superpolynomiální složitost
- Obvykle snadná garance správnosti algoritmu

## 2. Hltavý (greedy) algoritmus



- Očekávané řešení:  $(p_1, p_2, \dots, p_n)$
- Začít s  $\emptyset$
- Zafixovat vždy 1 položku řešení (např.  $p_2$ ), už zůstane neměnná, pak další položku (např.  $p_1$ ) atd., až vše pevně nastaveno
- Pořadí podle slibnosti položek (od slibnějších k méně)
- "greedy" – nikdy se nevrací, nikdy nemění dříve udělané rozhodnutí

- Př.: Vybrat nejatraktivnější zvířata do ZOO



- Některé položky se vzájemně vylučují - ale nedělá se žádná predikce, žádný návrat

- Realizace jedné cesty od kořene k uzlu v rozhodovacím stromě
- Obvykle velmi účinné algoritmy
- Greedy algoritmy – snadný vývoj i implementace
- Obvykle neumí najít globální extrém, ale poskytne dobrou heuristiku



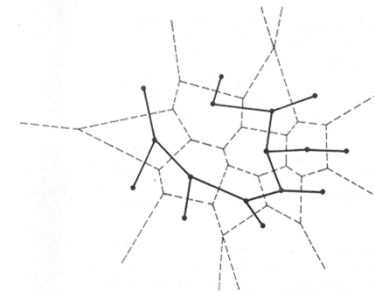
# Př.1: Greedy algoritmus pro minimální kostru (minimum spanning tree - MST)

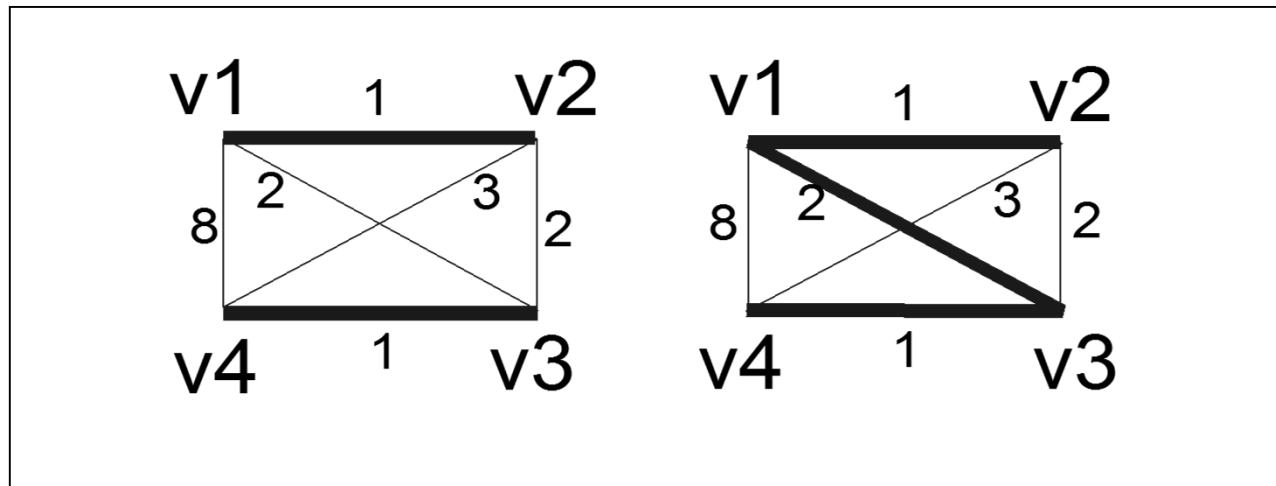
- Input: Ohodnocený spojitý graf

$$G=(V,E,c),c:E \rightarrow N$$

účinný alg.

- Output:  $(V,E')$ , kde  $E'$  - min. kostra
- Step 1: Seřad' hrany podle jejich ohodnocení:  
 $c(e_1) \leq c(e_2) \leq \dots \leq c(e_m)$
- Step 2: Set  $E' := \{e_1, e_2\}, i := 3;$
- Step 3: **while**  $|E'| < |V| - 1$  **do**
  - **begin**
  - Přidej  $e_i$  do  $E'$  pokud  
 $(V, E' \cup \{e_i\})$  neobsahuje smyčku;
  - $i := i + 1$
  - **end**





- Jaká je složitost nejhoršího případu vzhledem k počtu hran  $|E|$  ?
- Jak implementovat krok 3 v  $O(1)$  nebo  $O(\log n)$  na jednu hranu? (Př. řešení: Union & Find)

## Př.2: Greedy heuristika pro obchodního cestujícího (travelling salesperson - TSP)

- Input: Ohodnocený úplný graf

$$G=(V,E,c), c:E \rightarrow N, |V|=n, n \in \mathbb{N}^+$$

velmi slabé

- Output:  $(V, E')$ , kde  $E'$  - min. cesta

- Step 1: Seřad' hrany podle jejich ohodnocení:

$$c(e_1) \leq c(e_2) \leq \dots \leq c(e_m), m = \binom{n}{2}$$

- Step 2: Set  $E' := \{e_1, e_2\}, i := 3;$

- Step 3: **while**  $|E'| < n$  **do begin**

- Přidej  $e_i$  do  $E'$  pokud

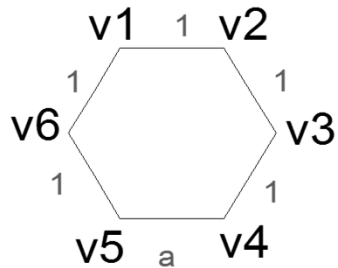
$(V, E' \cup \{e_i\})$  neobsahuje vrchol st.  $> 2$

ani smyčku délky kratší než  $n$ ;

- $i := i + 1$

- **end**

- Nalezené řešení může být jakkoliv horší než optimum (žádná zaručená mez kvality)

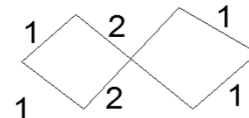


ostatní hrany váhu 2

$a \gg 1$

Greedy-TSP vezme  
napřed hrany délky 1  
 $\Rightarrow$  délka  $a+5$

Optimum: délka 8



### 3. Společné vlastnosti inkrementálních algoritmů

---

- Obvykle nevede k optimálním algoritmům
- Obvykle implementačně jednoduché, zejména vkládání
- Typicky využíváno pro úlohy v geometrickém modelování (budování nějaké struktury – např. triangulace nebo jiná dělení prostoru, průniky), v grafových úlohách nebo pro tvorbu datových struktur

## 4. Inkrementální vkládání

---

- Budujeme řešení (strukturu) o  $n$  položkách - napřed řešení (struktury) o  $n-1$  položkách, pak nezbytné změny a vložení  $n$ -té položky
- V paměti někdy dílčí řešení, někdy jen současný stav
- Lze využít také pro on-line problémy - vstup pak není celý dostupný na začátku – užitečné, ale část práce se dělá zbytečně
- Možné využití vah pro jednotlivé prvky
- Vhodné pro inovování existujícího řešení – doplnění nových prvků
- Někdy zapotřebí také rušení některých prvků (to už ovšem není „inkrementální metoda“) - typicky zapeklitý problém
- Široce uplatnitelná a modifikovatelná algoritmická metoda

## Př.5: Insertion\_sort

---

- Input: pole neseřazených hodnot  $A[1..n-1]$
- Output: seřazené pole  $A$
- Step 1:  $A[0] := -\infty$  // "zarážka (sentinel)"
- Step 2: **for**  $i := 1$  **to**  $n-1$  **do**
  - **begin**
  - $j := i;$
  - **while**  $(A[j] < A[j-1])$  **do begin**
    - $\text{Swap}(A[j], A[j-1]); j := j-1$  **end;**
  - **end**
- $O(n^2)$  v nejhor. př., pokud data skoro seřazena, podstatně lepší

Př.:

n=5

$-\infty, 1, 5, 2, 4$

i	j			
1	1	$1 < -\infty$	F	
2	2	$5 < 1$	F	
3	3	$2 < 5$	T	Swap(5,2)
3	2	$2 < 1$	F	
4	4	$4 < 5$	T	Swap(5,4)
4	3	$4 < 2$	F	

$-\infty, 1, 2, 5, 4$

$-\infty, 1, 2, 4, 5$

## Př.6: Obarvení vrcholů

---

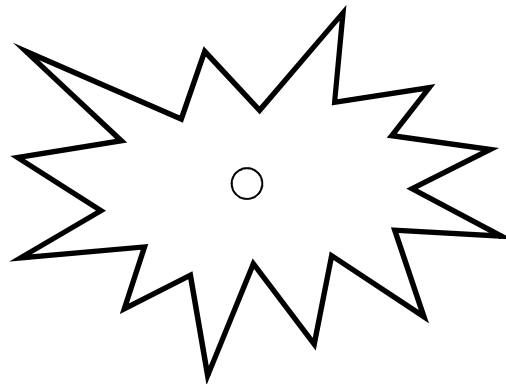
- Zadání: Obarvěte vrcholy grafu minimálním počtem barev tak, aby žádná hrana nespojovala vrcholy stejné barvy
- Aplikace: shlukování, plánování, optimalizace překladačů (užití koneč. počtu registrů - proměnné překrývající se v čase nemohou být ve stejném registru)
- Počet potřebných barev grafu - chromatické číslo, jeho určení je NP-úplný problém, pro planár.graf nejvýše 4
- => pro přesné určení pro malé grafy backtracking - pro malé náhodné grafy funguje překvapivě dobře <sup>14</sup>



- Heuristika: obarvovat vrcholy sekvenčně podle omezení daných už obarvenými
- Pořadí vrcholů: vkládat vrcholy v pořadí podle jejich stupně (sestupně)  $\leq$  vyšší stupeň- více omezení
- Vylepšení: výměna barev - zrušíme v obarvení všechny vrcholy kromě červených a modrých, pokud se rozpadne na 2 a více komponent, lze přebarvit červenou na modrou (nebo naopak ;- ) a pokračovat v barvení  $\Rightarrow$  růst kvality řešení za cenu růstu doby výpočtu a složitosti implementace

## Př.7: Konstrukce hvězdicového polygonu

- Zadání: sestrojte z množiny bodů  $S$  hvězdicový polygon  $P$

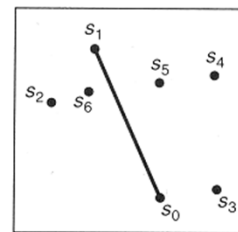


(Pozn.: pro danou  $S$  existuje více řešení, stačí nám jedno)

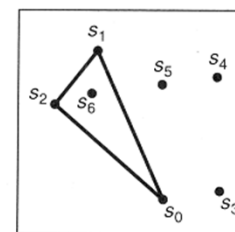
## Algoritmus: Inkrementální vkládání bodů z $S$ do $P$

- Start:  $P = (s_0, s_1, s_2)$  nebo  $(s_0, s_2, s_1)$  podle požadované orientace  $P$
- Vhodné místo pro další bod  $s_i, i=2, \dots, n-1$ :
  - Procházejí se hrany  $P$  např. po směru hodinových ručiček
  - $s_i$  se připojí

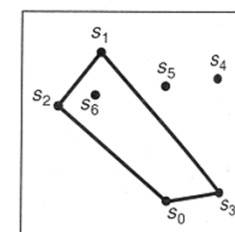
mezi vrcholy  
aktuální hrany,  
pokud jeho spojnice  
s těmito vrcholy  
nic neprotíná



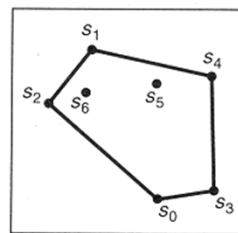
(a)



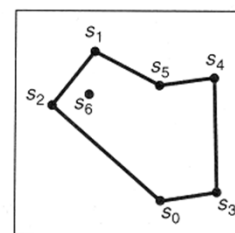
(b)



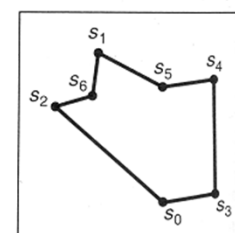
(c)



(d)



(e)



(f)

$O(n^2)$

## 5. Inkrementální konstrukce (inkrementální výběr)



- Budujeme řešení (strukturu) inkrementálně, v každém okamžiku "kousek,,
- Na rozdíl od vkládání se vytvořená část řešení už nezmění
- Vstup zpracováván v pořadí, jaké algoritmus chce, nikoliv v pořadí, jak vstup. data přicházejí - off-line - vstup musí být celý k dispozici
- Vhodné pořadí vstup. dat někdy určeno předem (např. presort), někdy až v průběhu výpočtu

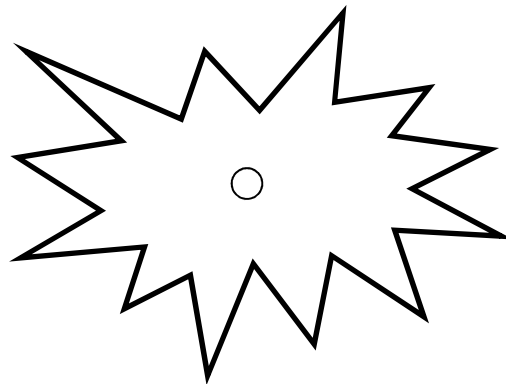
## Př.8: Selection\_sort

inkrementální konstrukce

- Algoritmus: opakovaně vybírá nejmenší položku z množiny, až je množina prázdná (v poli: vyměnit min s 1. prvkem atd.)
- $O(n^2)$ , ale jen  $n$  výměn - insertion\_sort v nejhorším případě kolem  $n^2/2$  polovýměn (posunů)

## Př.9: Konstrukce hvězdicového polygonu

- Zadání: sestrojte z množiny bodů  $S$  hvězdicový polygon  $P$

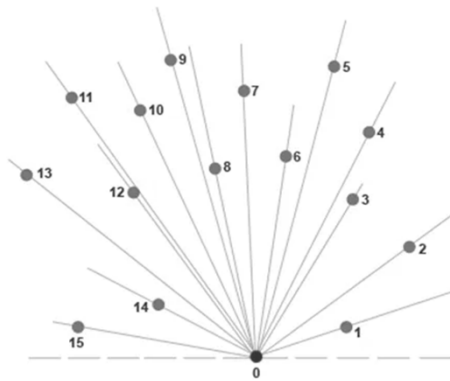


(Pozn.: pro danou  $S$  existuje více řešení, stačí nám jedno)

## Algoritmus: Inkrementální konstrukce polygonu $P$ z množiny bodů $S$

- Vybereme jeden z vrcholů a označíme  $c$
- Body  $S$  seřadíme vůči  $c$  úhlově
- Seřazené body v získaném pořadí pospojujeme

Volba  $c$ : centroid/kterýkoliv bod/extrém



$O(n \log n)$

# Literatura

---

- Literatura dostupná až pro konkrétní problémy