

## Hladové algoritmy

### Zalamování řádků v odstavci s lineární penalizací

Mějme text skládající se z  $n$  slov, šířka  $i$ tého slova je  $w_i$  pixelů. Chceme text vypsát do odstavce tak, aby byl zarovnaný do bloku a na každém řádku byly co nejmenší mezery (mezery musí být minimálně 1 pixel). Maximální délka řádku je  $m$  pixelů. Každému řádku kromě posledního je určena penalizace

$$p_{i,j} = (m - j + i - \sum_{k=i}^j w_k)$$

kde  $i$  a  $j$  jsou indexy prvního a posledního slova na řádku. Cílem je minimalizovat součet penalizací všech řádků (kromě posledního).

- ▶ Na rozdíl od minulého demonstračního cvičení je tentokrát penalizace lineární místo kubické. Dokážeme v tomto případě najít hladový algoritmus?

*Nezapomeňte:* Nejdůležitější součástí hladového algoritmu je důkaz jeho korektnosti.

# Hladové algoritmy

## Optimální bezprefixové kódování

Kódování množiny symbolů  $\Sigma$  je funkce  $code : \Sigma \rightarrow \{0, 1\}^*$ .

Kódování je *bezprefixové*, pokud žádný  $code(u)$  není prefixem  $code(v)$  pro žádná  $u \neq v$ . Dejme tomu, že známe četnosti všech symbolů v textu. Chceme najít takové bezprefixové kódování těchto symbolů, které bude minimalizovat celkovou délku zakódovaného textu.

- ▶ Jak souvisí bezprefixové kódování s binárními stromy?

# Hladové algoritmy

## Optimální bezprefixové kódování

Kódování množiny symbolů  $\Sigma$  je funkce  $code : \Sigma \rightarrow \{0, 1\}^*$ .

Kódování je *bezprefixové*, pokud žádný  $code(u)$  není prefixem  $code(v)$  pro žádná  $u \neq v$ . Dejme tomu, že známe četnosti všech symbolů v textu. Chceme najít takové bezprefixové kódování těchto symbolů, které bude minimalizovat celkovou délku zakódovaného textu.

- ▶ Jak souvisí bezprefixové kódování s binárními stromy?
- ▶ Jaký tvar musí mít každý binární strom, který reprezentuje optimální bezprefixové kódování?

# Hladové algoritmy

## Optimální bezprefixové kódování

Kódování množiny symbolů  $\Sigma$  je funkce  $code : \Sigma \rightarrow \{0, 1\}^*$ .

Kódování je *bezprefixové*, pokud žádný  $code(u)$  není prefixem  $code(v)$  pro žádná  $u \neq v$ . Dejme tomu, že známe četnosti všech symbolů v textu. Chceme najít takové bezprefixové kódování těchto symbolů, které bude minimalizovat celkovou délku zakódovaného textu.

- ▶ Jak souvisí bezprefixové kódování s binárními stromy?
- ▶ Jaký tvar musí mít každý binární strom, který reprezentuje optimální bezprefixové kódování?
- ▶ Má tento problém optimální substrukturu? Jak vypadají podproblémy?

# Hladové algoritmy

## Optimální bezprefixové kódování

Kódování množiny symbolů  $\Sigma$  je funkce  $code : \Sigma \rightarrow \{0, 1\}^*$ .

Kódování je *bezprefixové*, pokud žádný  $code(u)$  není prefixem  $code(v)$  pro žádná  $u \neq v$ . Dejme tomu, že známe četnosti všech symbolů v textu. Chceme najít takové bezprefixové kódování těchto symbolů, které bude minimalizovat celkovou délku zakódovaného textu.

- ▶ Jak souvisí bezprefixové kódování s binárními stromy?
- ▶ Jaký tvar musí mít každý binární strom, který reprezentuje optimální bezprefixové kódování?
- ▶ Má tento problém optimální substrukturu? Jak vypadají podproblémy?
- ▶ Navrhněte hladový přístup a *dokažte*, že je korektní.

# Hladové algoritmy

## Protínání intervalů

Mějme množinu  $\mathcal{I}$  (uzavřených) intervalů na číselné ose; intervaly mohou být různých délek a jsou zadány svými levými a pravými krajními body. O množině čísel  $X$  řekneme, že *protíná* množinu intervalů  $\mathcal{I}$ , pokud každý interval z  $\mathcal{I}$  obsahuje alespoň jedno číslo z  $X$ . Chceme najít minimální množinu  $X$ , která protíná  $\mathcal{I}$ .

- ▶ Navrhněte hladový přístup a *dokažte*, že je korektní.

# Hladové algoritmy

## Barvení intervalů

Mějme opět množinu (uzavřených) intervalů  $\mathcal{I}$ . *Nekonfliktní obarvení* intervalů z  $\mathcal{I}$  přiřadí každému intervalu barvu tak, že žádné dva intervaly se stejnou barvou nemají společný průnik. Chceme najít nekonfliktní obarvení s co nejmenším počtem barev.

- ▶ Navrhněte hladový přístup a *dokažte*, že je korektní.

# Backtracking

**Všechny ktice** (variace s opakováním)

Mějme dána přirozená čísla  $n$  a  $k$ . Napište pseudokód (rekurzivního) algoritmu, který vypíše na výstup všechny *ktice* čísel z  $\{1, 2, \dots, n\}$ .



# Backtracking

## Permutace

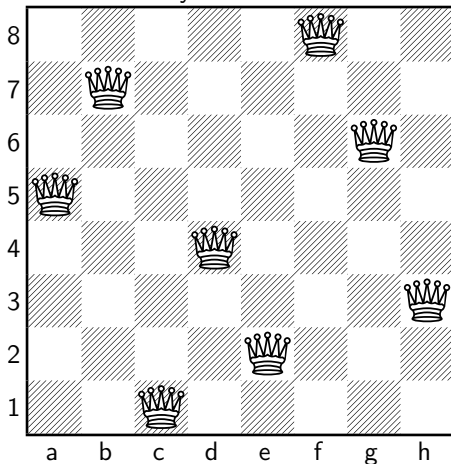
Mějme dáno přirozené číslo  $n$ . Máme napsat algoritmus, který vypíše všechny permutace čísel z  $\{1, 2, \dots, n\}$ .

- ▶ Jak můžeme využít algoritmus z minulého slajdu?

# Backtracking

## Problém $n$ královen

Mějme dáno přirozené číslo  $n$ . Chceme umístit  $n$  šachových figur královen na šachovnici o rozměrech  $n \times n$  tak, aby se žádné dvě z nich vzájemně neohrožovaly.



- Jak nám pomůže algoritmus z minulého slajdu?

# Backtracking

## **Kombinace** (bez opakování)

Mějme dána přirozená čísla  $n$  a  $k$ . Chceme vypsát všechny  $k$ prvkové kombinace čísel z  $\{1, 2, \dots, n\}$ , tj. všechny  $k$ prvkové podmnožiny této množiny.

# Backtracking

## Klika

Mějme neorientovaný graf  $G = (V, E)$ . *Klikou v grafu* nazveme každou podmnožinu  $K \subseteq V$  takovou, že podgraf indukovaný vrcholy z  $K$  je úplný.

- ▶ Mějme zadáno číslo  $k$ . Chceme zjistit, jestli  $G$  obsahuje kliku velikosti  $k$ . Jak můžeme modifikovat některý z dříve uvedených algoritmů?

# Backtracking

## Klika

Mějme neorientovaný graf  $G = (V, E)$ . *Klikou v grafu* nazveme každou podmnožinu  $K \subseteq V$  takovou, že podgraf indukovaný vrcholy z  $K$  je úplný.

- ▶ Mějme zadáno číslo  $k$ . Chceme zjistit, jestli  $G$  obsahuje kliku velikosti  $k$ . Jak můžeme modifikovat některý z dříve uvedených algoritmů?
- ▶ Chceme zjistit, jakou největší kliku  $G$  obsahuje. Jak můžeme modifikovat předchozí algoritmus tak, abychom dostali co nejefektivnější řešení?