

# Backtracking

## Motivačný príklad

**Problém  $n$  dám** - na šachovnicu s  $n \times n$  políčkami je treba rozložiť  $n$  dám tak, aby sa neohrozovali (dámy sa ohrozuju ak ležia v jednom riadku, jednom stĺpci, alebo na jednej uhlopriečke).

- Drevorubačské riešenie: vyskúšať všetky možnosti - ( $n^2$  nad  $n$ )
- **Backtracking**: postupné pridávanie dám (najprv sa umiestni prvá dáma, potom druhá, ...) tak, aby sa neohrozovali. V prípade, že k  $m$  ( $m \leq n$ ) už rozloženým dámam nevieme pridať ďalšiu tak, aby sa neohrozovali, musíme sa vrátiť (backtrack) k  $m$ -tej dáme a umiestniť ju na iné miesto, pričom však stále musí platiť, že sa dámy neohrozuju. Ak je to možné, pokračuje sa ďalej, pridáva sa ďalšia,  $(m+1)$ -vá dáma. Ak nie je možné takú pozíciu pre  $m$ -tú dámu nájsť, je nutné vrátiť sa k  $(m-1)$ -vej dáme a premiestniť tú...

## Backtracking pre problém $n$ dám

- Aby sa dámy neohrozovali, žiadne 2 nemôžu byť v jednom stĺpci a teda v každom stĺpci je práve jedna dáma. Budeme ich preto umiestňovať vždy  $m$ -tú dámu do  $m$ -tého stĺpca.
- Rozloženie  $m$  dám je *prípustné* ( $m \leq n$ ), ak všetkých  $m$  dám rozložených na šachovnici sa neohrozuje (žiadne 2 dámy neležia v jednom riadku, stĺpci, na jednej uhlopriečke).
- Ak rozloženie dám nie je prípustné, nastáva *kolízia*.

Ukážka pre  $n=4$ :

o			

Uložíme prvú dámu do prvého stĺpca a prvého riadku a prejdeme na druhú dámu.

o	o		

Druhú dámu sa snažíme umiestniť do druhého stĺpca a prvého riadku, ale tam ju nemôžeme uložiť, lebo by sa 1. a 2. dáma ohrozovali (sú v jednom riadku), nastala by kolízia.

o			
	o		

Podobne, druhá dáma nemôže byť v druhom riadku (dámy by sa ohrozovali po uhlopriečke) - kolízia.

o			

	o		

Druhú dámu preto môžeme umiestniť až do tretieho riadku (rozloženie dám je prípustné).

o		o	
	o		

Keď sme umiestnili druhú dámu, pokračujeme s treťou (v treťom stĺpci). Nemôže byť v prvom riadku (kolízia - v prvom riadku už je prvá dáma),

o			
		o	
	o		

nemôže byť v druhom riadku (kolízia - bola by na jednej uhlopriečke s druhou dámou),

o			
	o	o	

nemôže byť v treťom riadku (kolízia - bola by v jednom riadku s druhou dámou) a

o			
	o		
		o	

nemôže byť ani vo štvrtom riadku (kolízia - bola by na jednej uhlopriečke s druhou dámou).

o			
	o		

Preto všetky možnosti, kde je usporiadanie prvej a druhej dámy takéto, zahodíme a *späťne sa vrátíme* (*backtrack*) k druhej dámě a skúsime ju dať do ďalšieho (štvrtého) riadku. Rozloženie je prípustné, preto

pokračujeme s treťou dámou v treťom stĺpci.

o		o	
	o		

Kolízia (riadok).

o			
		o	
	o		

Rozloženie je prípustné, pokračovanie so štvrtou dámou vo štvrtom stĺpci.

o			o
		o	
	o		

Kolízia (riadok, uhlopriečka).

o			
		o	o
	o		

Kolízia (riadok).

o			
		o	
			o
	o		

Kolízia (uhlopriečka).

o			
		o	

	o		o
--	---	--	---

Kolízia, pre štvrtú dámu (riadok) už nie je žiadna možnosť => *backtrack*.

o			
		o	
	o		

Kolízia (uhlopriečka).

o			
	o	o	

Kolízia, pre tretiu dámu (riadok) už nie je žiadna možnosť => *backtrack*, podobne aj druhá dáma je v poslednom riadku a nemá už ďalšie možnosti => *backtrack* - prvú dámu do druhého riadku.

o			

Rozloženie je prípustné, pokračujeme s druhou dámou.

	o		
o			

Kolízia (uhlopriečka).

o	o		

Kolízia (riadok).

o			
	o		

Kolízia (uhlopriečka).

o			
	o		

Rozloženie je prípustné, pokračujeme s treťou dámou.

		o	
o			
	o		

Rozloženie je prípustné, pokračujeme so štvrtou dámou.

		o	o
o			
	o		

Kolízia (riadok).

		o	
o			o
	o		

Kolízia (uhlopriečka).

		o	
o			
			o
	o		

Našli sme rozloženie 4-och dám.

- Tu sme vyskúšali 26 možností.
- Pri drevorubačskom spôsobe by to bolo  $(4^2)! / (4! * 12!) = 1820$  možností.

## Schéma algoritmu pre problém n dám

```
POLOZ_DAMU(stlpec)
  for riadok <- 1 to n
    if umiestnenie damy na poziciu [riadok, stlpec]
      je pripustne
      then zaznac damu na poziciu [riadok, stlpec]
        if (stlpec = n)
          then VYPIS_RIESENIE
        else POLOZ_DAMU(stlpec+1)
      zrus damu z pozicie [riadok, stlpec]
```

začína sa s POLOZ\_DAMU(1)

## Všeobecná schéma pre backtracking

```
BACKTRACK(i)
  for všetky prípustné j
    zaznač_krok(i, j)
    if skončené
      then VYPIS_RIESENIE
    else BACKTRACK(i+1)
  zrus_krok(i, j)
```