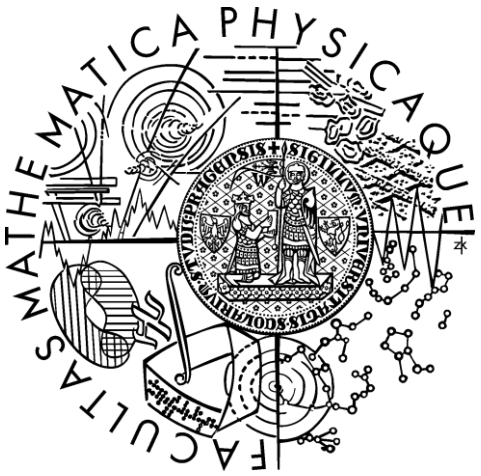


Vizualizace jako prostředek k získání znalostí o kvalitě řešení problémů pohybu po grafu

Pavel Surynek a **Petr Koupý**

Matematicko-fyzikální fakulta
Univerzita Karlova v Praze

Problémy pohybu po grafu

- **Abstrakce** k úlohám o pohybu více (autonomních nebo pasivních) entit v jistém prostředí (fyzickém či virtuálním).
  - Entity mají dané **počáteční** a **cílové** rozmístění v prostředí.
  - Chceme **naplánovat pohyby entit v čase**, aby dosáhly cílového rozmístění a přitom **respektovaly fyzikální omezení**.
- **Fyzikální omezení** jsou:
  - Entity spolu **nesmějí kolidovat**.
  - Entity **nesmějí narážet do překážek** vyskytujících se prostředí.
- Základní **abstrakce** k úloze o pohybu:
  - Úloha o **pohybu kamenů po grafu** (*pebble motion on a graph*).
  - Existují i jiné abstrakce – typicky ovšem **převoditelné** na úlohu o kamenech.

Úloha o **pohybu kamenů po grafu** (1)

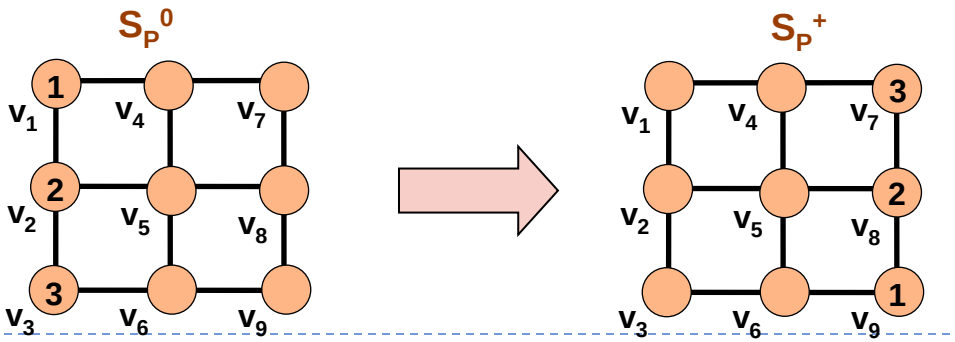
Wilson, 1974; Kornhauser et al., 1984

- Populární pohybová úloha, jež lze abstrahovat jako problém pohybu kamenů po grafu je známá jako **Lloydova patnáctka (devítka).**
  - **Entita** je **kámen** s číslem (*pebble*).
- Prostředí modelujeme jako neorientovaný graf, kde **vrcholy reprezentují pozice** v prostředí a **hrany možnost průchodu entity mezi pozicemi**.
- **Formální definice** problému pohybu kamenů po grafu:
  - Je to čtveřice $\Pi = (G, P, S_P^0, S_P^+)$, kde:
    - $G=(V,E)$ je neorientovaný **graf**,
    - $P = \{p_1,p_2,...,p_\mu\}$, kde $\mu<|V|$ je **množina kamenů**,
    - S_P^0: $P \rightarrow V$ je prostá funkce určující **počáteční rozložení** kamenů a
    - S_P^+: $P \rightarrow V$ je prostá funkce určující **cílové rozložení** kamenů.

Úloha o **pohybu kamenů po grafu** (2)

Wilson, 1974; Kornhauser et al., 1984

- Čas modelujeme diskrétně. **Časové kroky** a jejich uspořádání jsou izomorfní struktuře přirozených čísel.
- **Dynamicita** úlohy je následující:
  - Kámen nacházející se v časovém kroku i v jistém vrcholu se může přesunout do sousedního vrcholu v časovém kroku i+1, jestliže cílový vrchol je v časovém kroku i **neobsazený** a **žádný jiný kámen** se současně nepřesouvá do stejného cílového vrcholu.
- Pro dané $\Pi = (G, P, S_P^0, S_P^+)$, chceme najít:
  - Posloupnost pohybů pro každý kámen tak, že je splněna podmínka dynamicity úlohy a kámen dosáhne cílového vrcholu.



Řešení problému pohybu po grafu, kde P={1,2,3}

délka řešení=7

$\mathbf{M_1}=[\mathbf{v_1}, v_4, v_7, v_8, v_9, v_9, \mathbf{v_9}]$
$\mathbf{M_2}=[\mathbf{v_2}, v_2, v_1, v_4, v_7, v_8, \mathbf{v_8}]$
$\mathbf{M_3}=[\mathbf{v_3}, v_3, v_3, v_2, v_1, v_4, \mathbf{v_7}]$

Časový krok: **1 2 3 4 5 6 7**

Má smysl řešit úlohy pohybu po grafu?

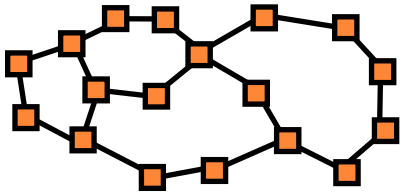
- Přesouvání kontejnerů
  (entita = **kontejner**)
- Intenzivní doprava
  (entita = **automobil** (v zácpě))
- Přesuny dat
  (entita = **datový paket**)
- Zobecněné výtahy
  (entita = **výtah**)

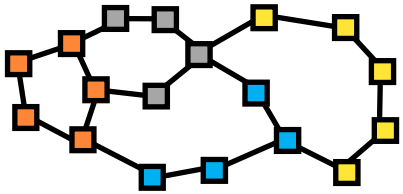
Je úloha pohybu po grafu **lehká** či **těžká**?

- **Základní** varianta úlohy je **snadno řešitelná**:
  - Existuje algoritmus pracující v polynomiálním čase ($O(|V|^3)$), který vygeneruje řešení délky $O(|V|^3)$ úlohy pohybu kamenů po grafu (Kornhauser et al., 1984).
- Chceme-li najít **řešení nejkratší možné délky** obtížnost roste:
  - Optimální varianta úloh pohybu po grafu je **NP-těžká** a navíc **neexistuje** pro ni **polynomiální aproximační schéma** (pokud P≠NP) (Ratner a Warmuth, 1986).
- Zaměřili jsme se na vygenerování **kvalitnějších** (tj. kratších) **neoptimálních** řešení:
  - Omezení na **2-souvislé grafy** - téměř vždy je úloha řešitelná.

Případ **2-souvislého** grafu

- Úlohy pohybu po grafu na **2-souvislých** grafech jsou z praktického hlediska nejdůležitější.
  - Téměř všechna cílová rozložení entit v grafu jsou **dosažitelná** z libovolného počátečního rozložení.
- **Další omezení** je, že dovolujeme pouze **jeden volný vrchol** (toto představuje neobtížnější situaci).
- Neorientovaný graf G=(V,E) je **2-souvislý**, jestliže $|V| \geq 3$ a $\forall v \in V$ je graf $G=(V-\{v\},E')$, kde $E'=\{\{x,y\} \in E \mid x,y \neq v\}$, souvislý.
- **Důležitá vlastnost:** Každý 2-souvislý graf sestrojit postupným přidáváním uch k počáteční kružnici → **rozklad na ucha**

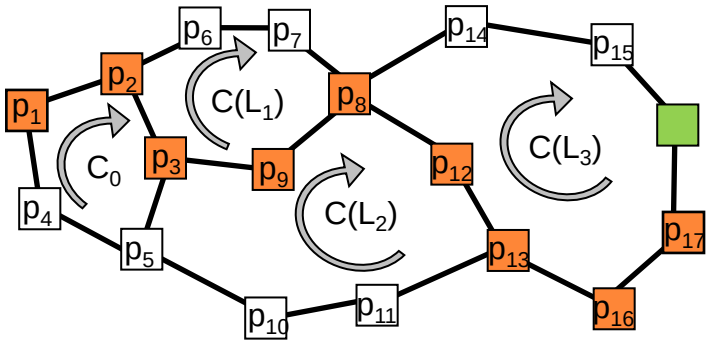




počáteční kružnice
1. ucho
2. ucho
3. ucho

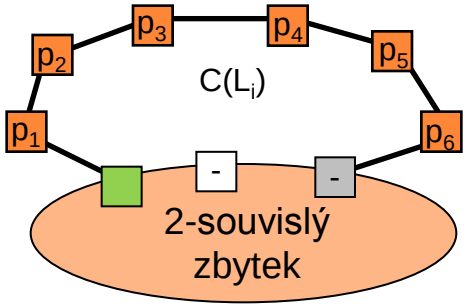
Algoritmus **BIBOX-θ** (1)

- Algoritmus **BIBOX** řeší úlohy pohybu kamenů po grafu.
  - Předpokládá, že prostředí je modelováno **2-souvislým** grafem.
    - Opírá se o **znalost rozkladu na ucha**.
  - Předpokládáme právě **jeden neobsazený** vrchol.
    - Není-li tomu tak, obsadí se vrcholy fiktivními kameny. Jejich pohyb se pak z finálního řešení odfiltruje.
  - Algoritmus opět pracuje v polynomiálním čase ($O(|V|^3)$), ovšem konstanta v odhadu je nižší než u algoritmu z (Kornhauser et al., 1984).
- Základní dovednost je přesun kamenu do vybraného vrcholu grafu:
  - **Přesouvání volného** vrcholu a
  - **rotace podél uch**.

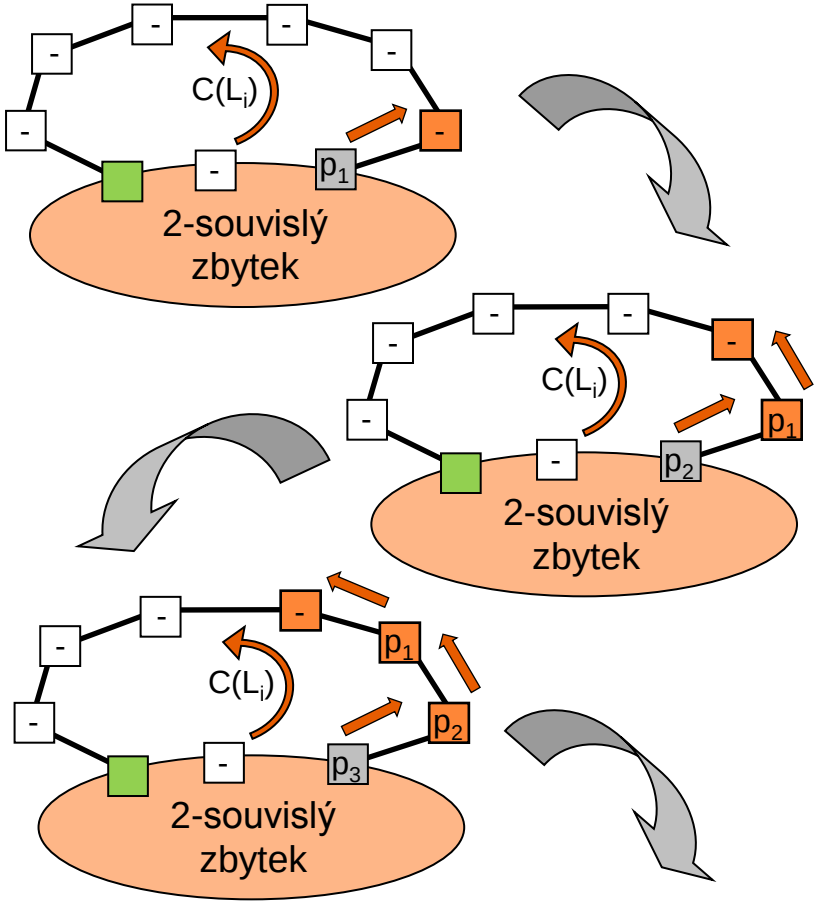


Algoritmus **BIBOX-θ** (2)

- Zvládáme-li přesun individuálního kamenu lze provádět složitější přesuny:
  - Skládání kamenů/robotů do ucha ve správném pořadí.

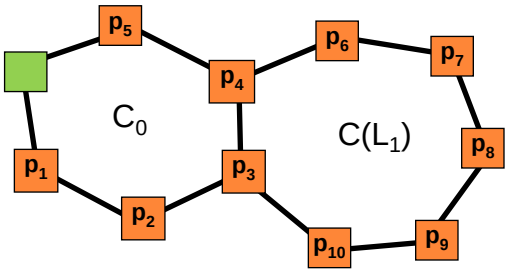


- **Zásobníkové skládání**:
  - Vezmeme **poslední ucho** rozkladu.
    - Přesuneme kámen do šedého vrcholu.
    - Provedeme rotaci ucha (pomocí **zeleného** volného vrcholu).
    - ...



Algoritmus **BIBOX-θ** (3)

- Počáteční kružnice a první ucho rozkladu (tzv. **θ-graf**) představují zvláštní situaci.
- Zásobníkové skládání, zde **nefunguje**.
- Výslednou (sudou) **permutaci** kamenů nutno poskládat z jednotlivých **rotací podél 3-cyklů** (bez detailů).

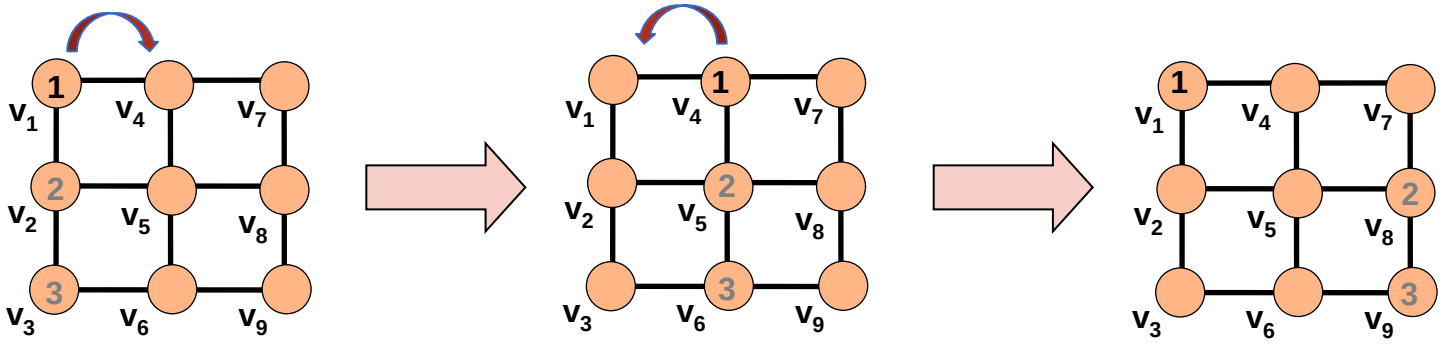


  - **Úzké hrdlo** algoritmu – (známé neoptimální) rotace podél 3-cyklů používají příliš mnoho pohybů.
  - Používá se **databáze** s předvypočtenými optimálními řešeními pro rotace podél 3-cyklů (obdoba databáze vzorů).
  - **Výsledné** (neoptimální) **řešení** poskládat z optimálních řešení pro rotace podél 3-cyklů.
    - → **Relativně kvalitní** neoptimální řešení.

Hlavní **nedostatek** popsaného přístupu

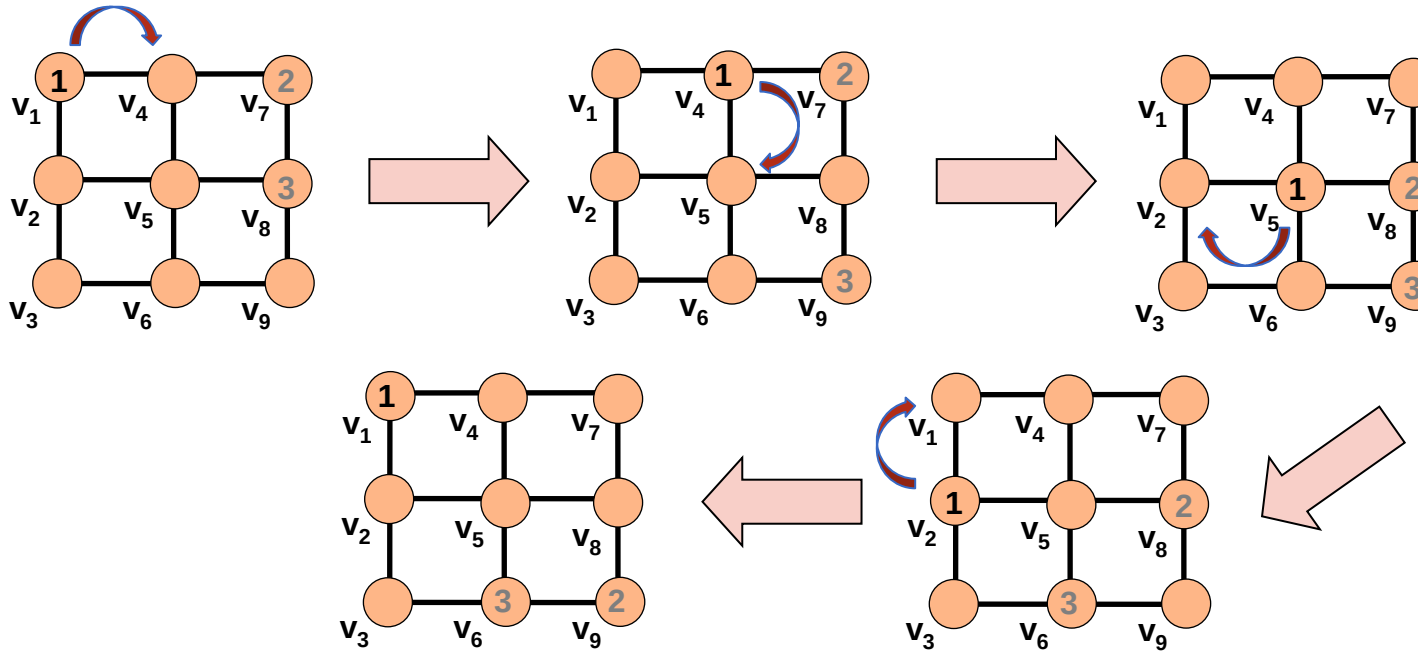
- Není-li graf zcela zaplněn kameny může navržený postup vytvářet jisté **redundance** v rámci řešení.
  - **Přidáme fiktivní** kameny, úlohu vyřešíme.
  - Z vygenerovaného řešení **odstraníme** pohyby **fiktivních** kamenů.
- Pomocí vizualizačního programu **GraphRec** se podařilo identifikovat několik typů **redundancí** v řešeních:
  - **(i) Inverzní pohyby**
    - Pohyb negující bezprostředně předcházející pohyb.
  - **(ii) Redundantní pohyby**
    - Sekvence pohybů, která kámen vrátí do výchozí pozice.
  - **(iii) Dlouhá posloupnost pohybů**
    - Sekvence pohybů, která kámen přesouvá do jiného vrcholu, ale přitom existuje kratší sekvence pohybů provádějící totéž.

(i) **Inverzní** pohyby



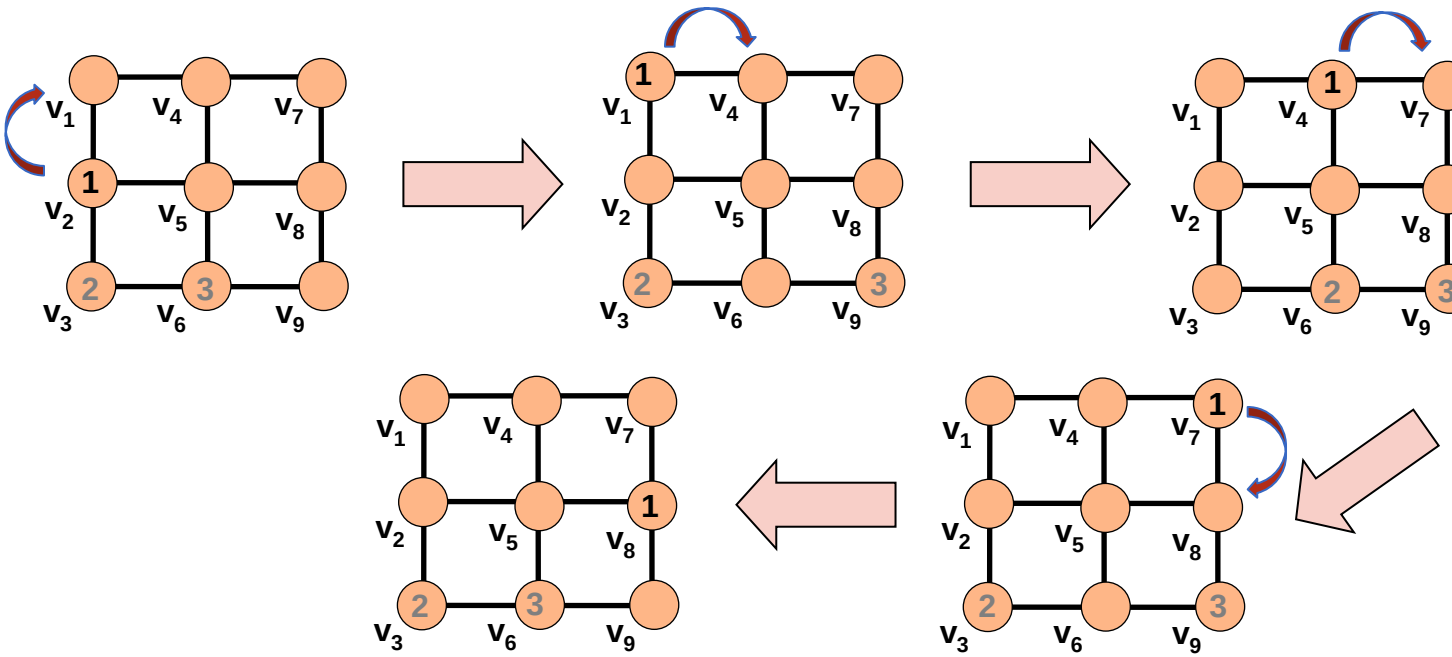
- **Kámen 1** provedl dvojici **inverzních pohybů**.
  - Mějme posloupnost pohybů Φ
  - Jednoduchým algoritmem lze inverzní pohyby odstranit v čase $O(|\Phi|^2)$
  - Odstranění inverzního pohybu může mít za následek vznik nových inverzních pohybů

(ii) **Redundantní** pohyby



- **Kámen 1** provedl posloupnost **redundantních pohybů.**
  - Bez interference s ostatními kameny návrat na výchozí pozici.
  - Jednoduchým algoritmem lze redundantní pohyby odstranit v čase $O(|\Phi|^4)$.
  - Rovněž mohou vznikat nové redundantní posloupnosti.

(iii) Dlouhá posloupnost pohybů



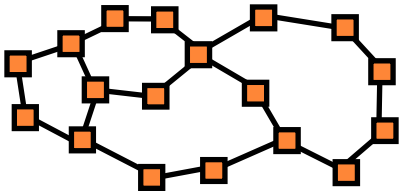
- **Kámen 1** provedl **dlouhou posloupnost** pohybů**.**
  - Bez interference s ostatními kameny možno jít kratší cestou.
  - Jednoduchým algoritmem lze redundantní pohyby odstranit v čase $O(|\Phi|^4+|\Phi|^3|V|^2)$.
  - Opět mohou vznikat nové dlouhé posloupnosti pohybů.

Vizualizace jako prostředek k získání znalostí o kvalitě řešení problémů pohybu po grafu

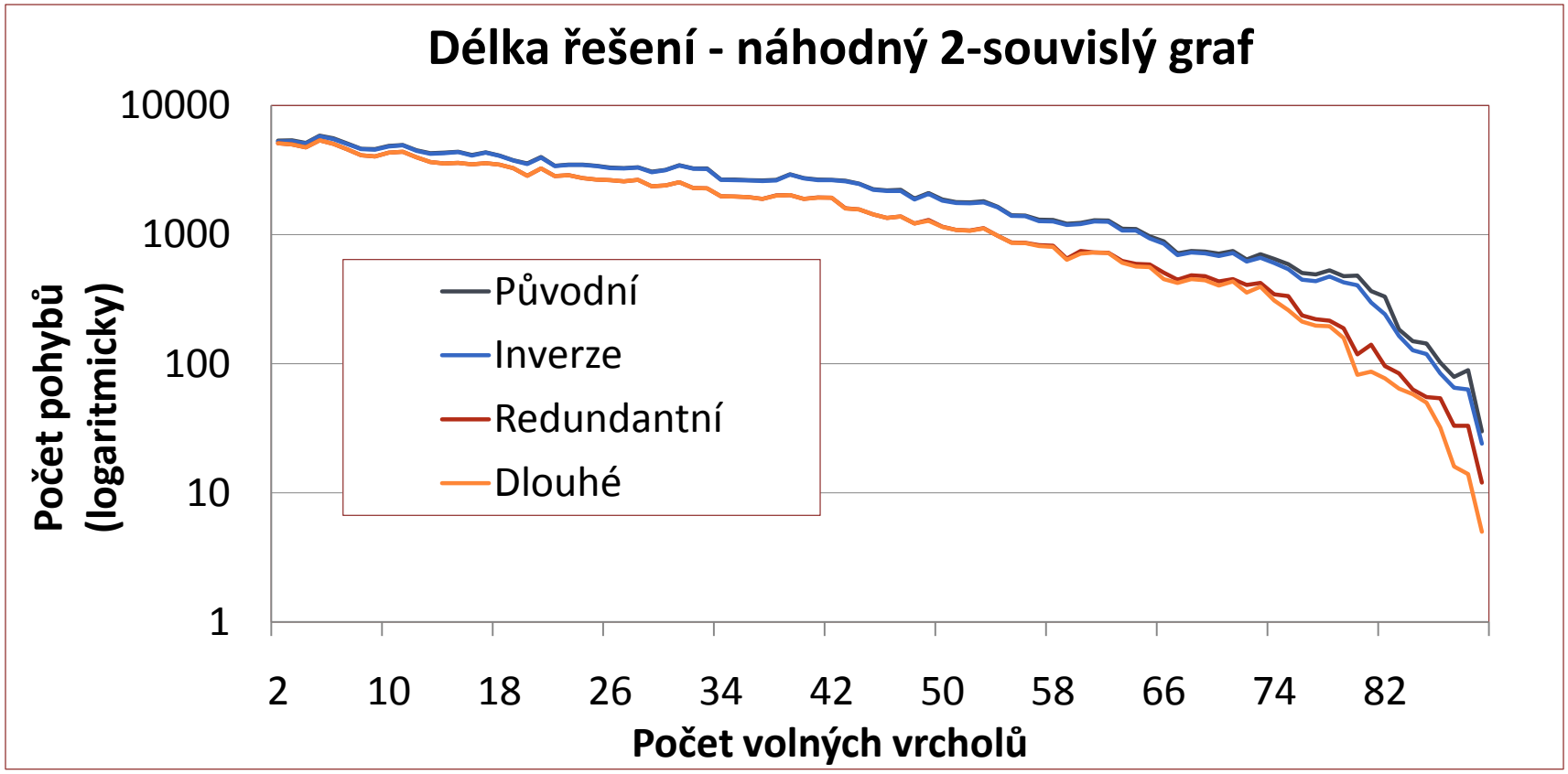
Pavel Surynek a **Petr Koupý**

Demonstrace programu GraphRec

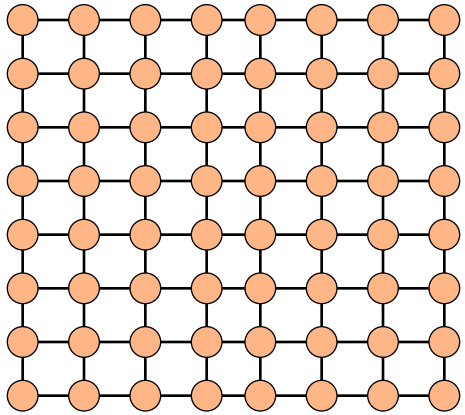
Experimenty (1)



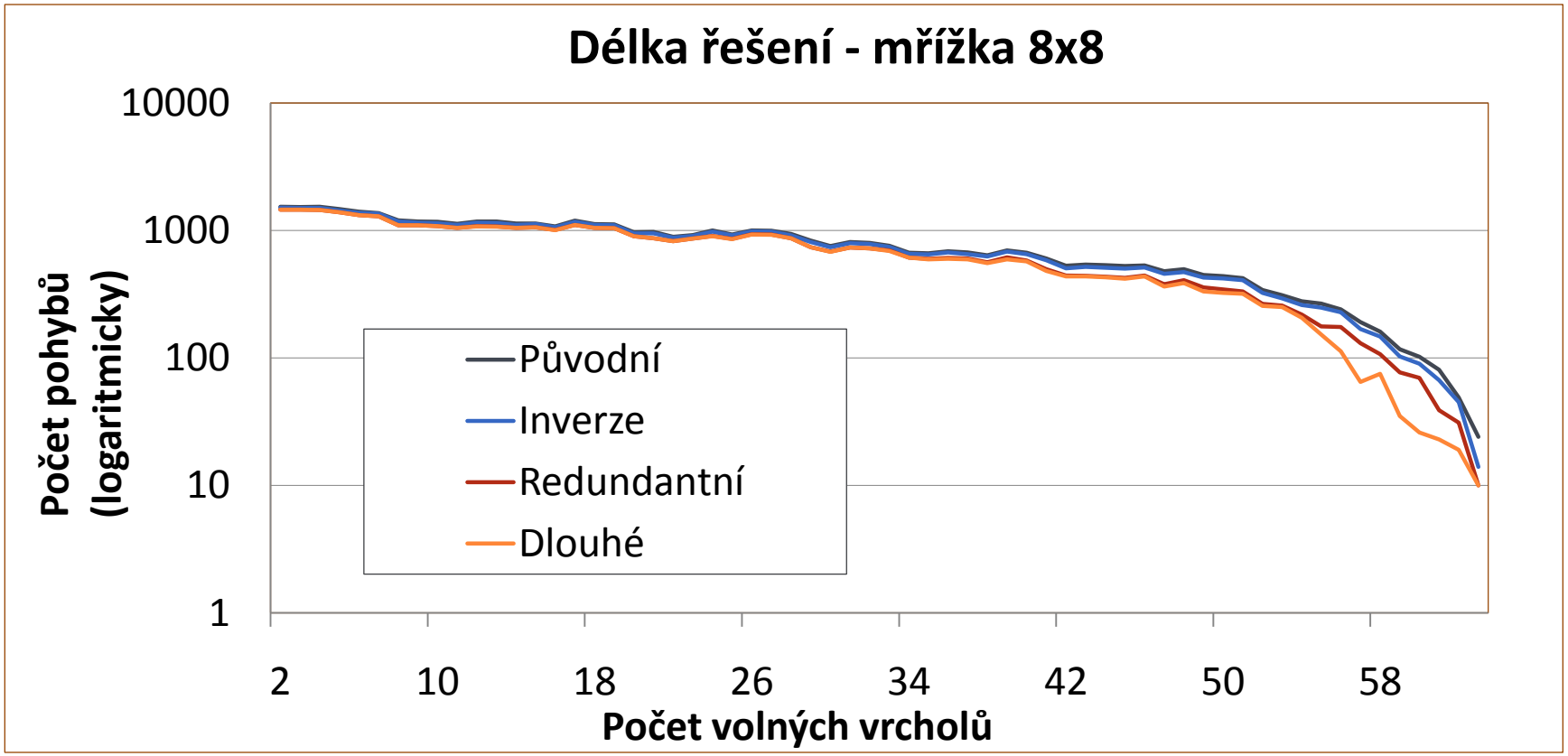
- **Náhodný 2-souvislý** graf:
  - Postupné **přidávání uch náhodné délky** k již zkonstruovanému grafu.
  - Počáteční a cílové rozložení kamenů voleno jako náhodná permutace.



Experimenty (2)



- **Mřížka 8x8**:
  - Opět náhodné počáteční a cílové rozložení kamenů opět voleno jako **náhodná permutace**.



Závěrečné poznámky

- Vizualizační program **GraphRec** umožnil **získat znalosti** o řešení problému pohybu po grafu.
- Na základě získaných znalostí **byly identifikovány redundance** v generovaných řešeních a byly navrženy opatření na jejich odstranění.
- Provedené experimenty ukázaly, že odstraněním redundancí **lze řešení podstatně zkrátit**, zvláště v případě, kdy graf obsahuje více volných vrcholů.