Stavový prostor
(state space)
a strategie pohybu v něm

- Stavový prostor, graf
- Strategie pohybu stavovým prostorem
- Prostor prohledávání

Problém batohu

Jsou dána přirozená čísla n, M, c_1, c_2, ..., c_n, w_1, w_2, ..., w_n. Nalezněte čísla x_1, x_2, ..., x_n z množiny {0,1} tak, aby

$$\sum_{i=1}^{n} x_i w_i \leq M \qquad \sum_{i=1}^{n} x_i c_i = \max.$$

Je dáno n věcí, i-tá věc má váhu w_i a cenu c_i, dále batoh s nosností M. Nalezněte takovou sestavu věcí v batohu, aby nebyl přetížen a cena věcí byla maximální.

Instance a konfigurace

instance n =3, M =6, C ={10, 20, 30}, W ={2, 3, 5}

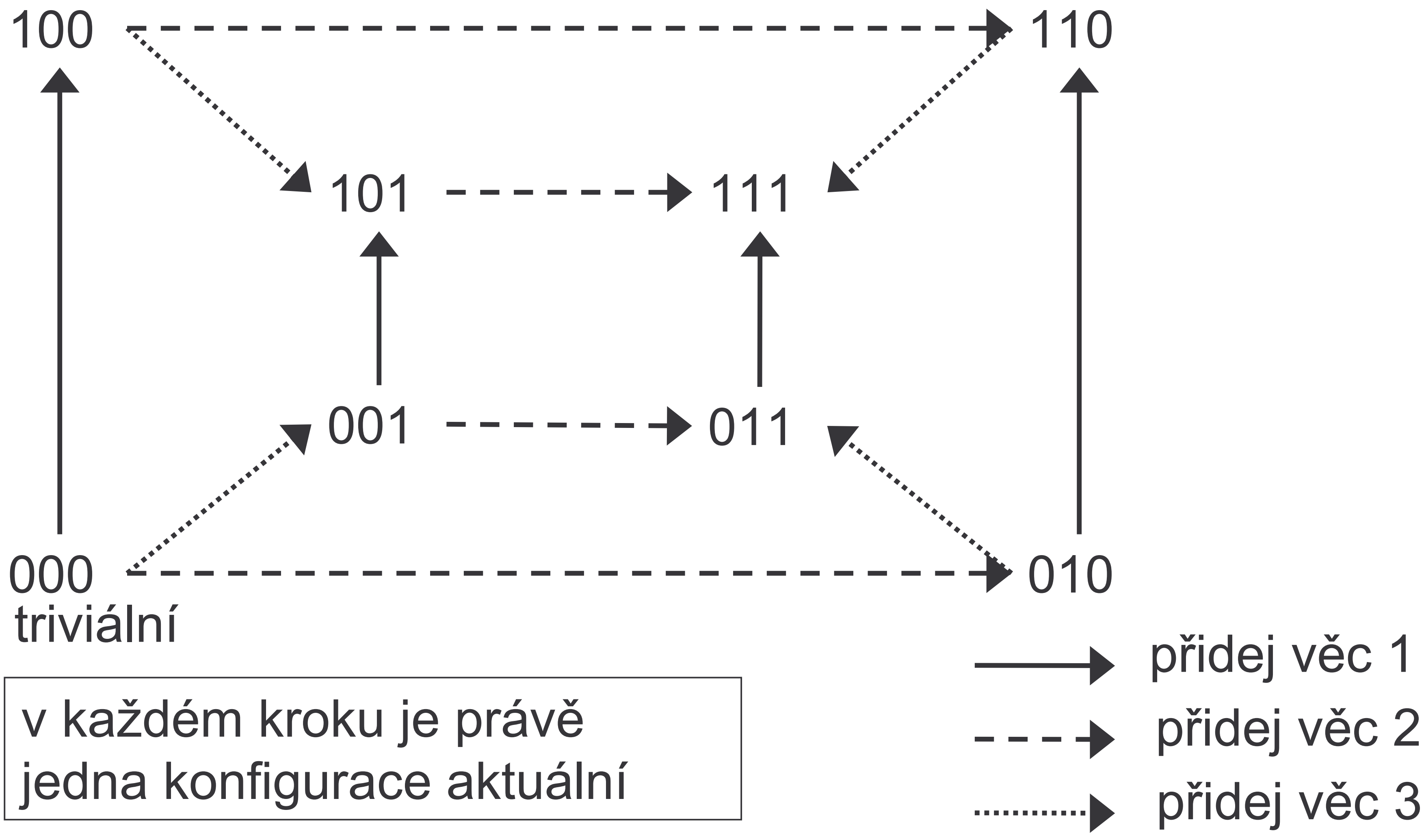
všechny konfigurace

x_1	x_2	x_3	$\Sigma x_i c_i$	$\Sigma x_i w_i$	
0	0	0	0	0	✓ triviální
0	0	1	30	5	✓ ✔
0	1	0	20	3	✓
0	1	1	50	8	⊘
1	0	0	10	2	✓
1	0	1	40	7	⊘
1	1	0	30	5	✓ ✔
1	1	1	60	10	⊘

řešení ✓

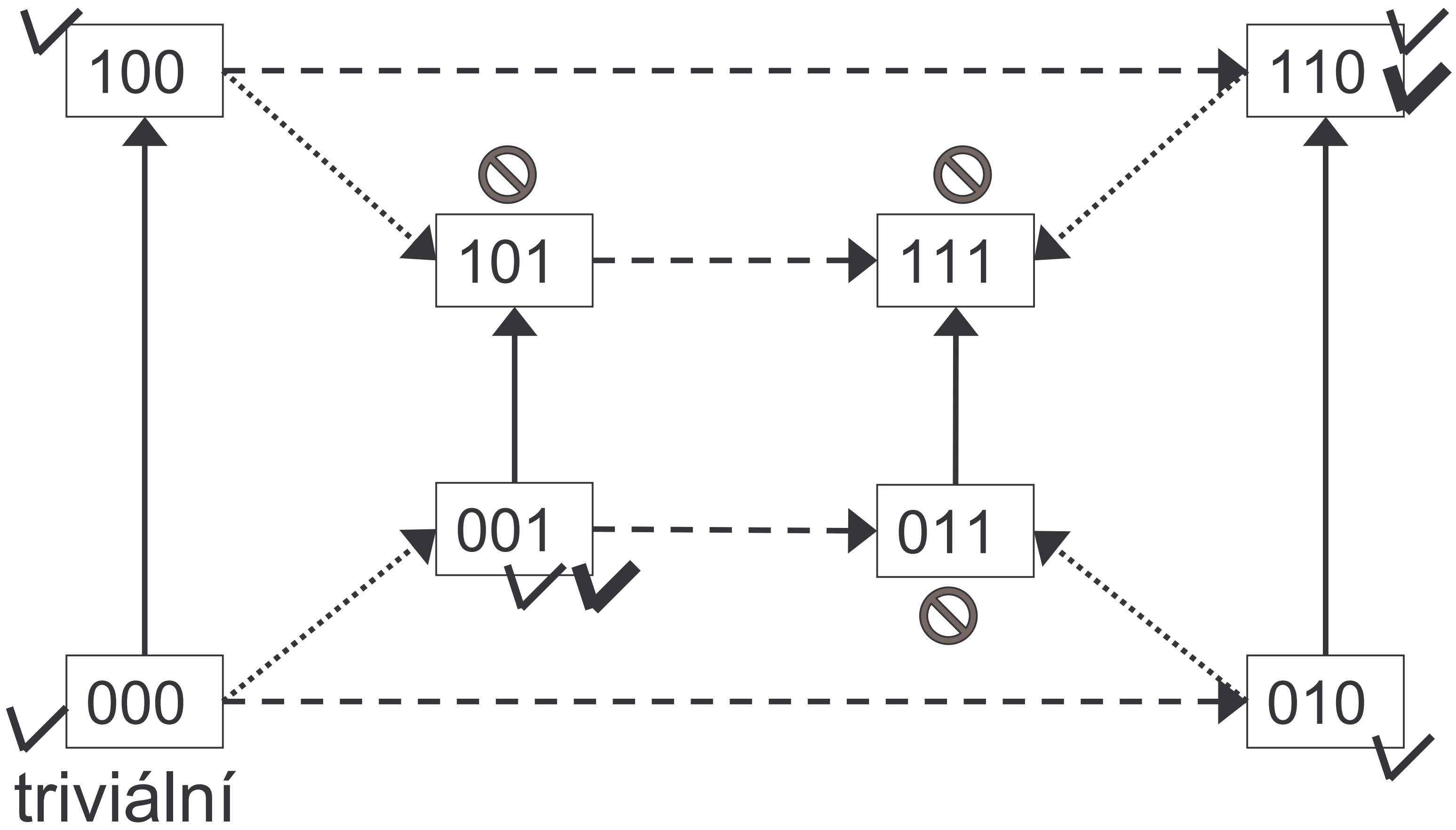
optimální řešení ✔

Průchod hladovým algoritmem

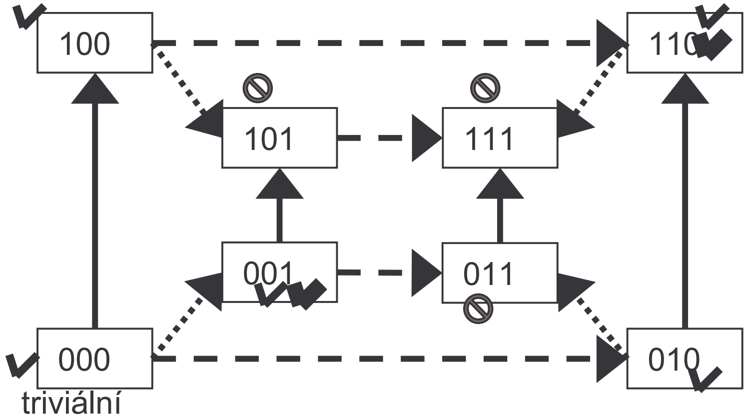


v každém kroku je právě
jedna konfigurace aktuální

Hodnocení konfigurací



Vlastnosti



- Každý uzel je konfigurace
- Vloženou věc nelze vyjmout
- V grafu nelze zabloudit
- Následník každé konfigurace, pokud existuje, má větší celkovou váhu i cenu
- Následník každé nepřípustné konfigurace je nepřípustná konfigurace
- Odráží charakteristiky problému i algoritmu, který jej řeší

Stavový prostor
state space

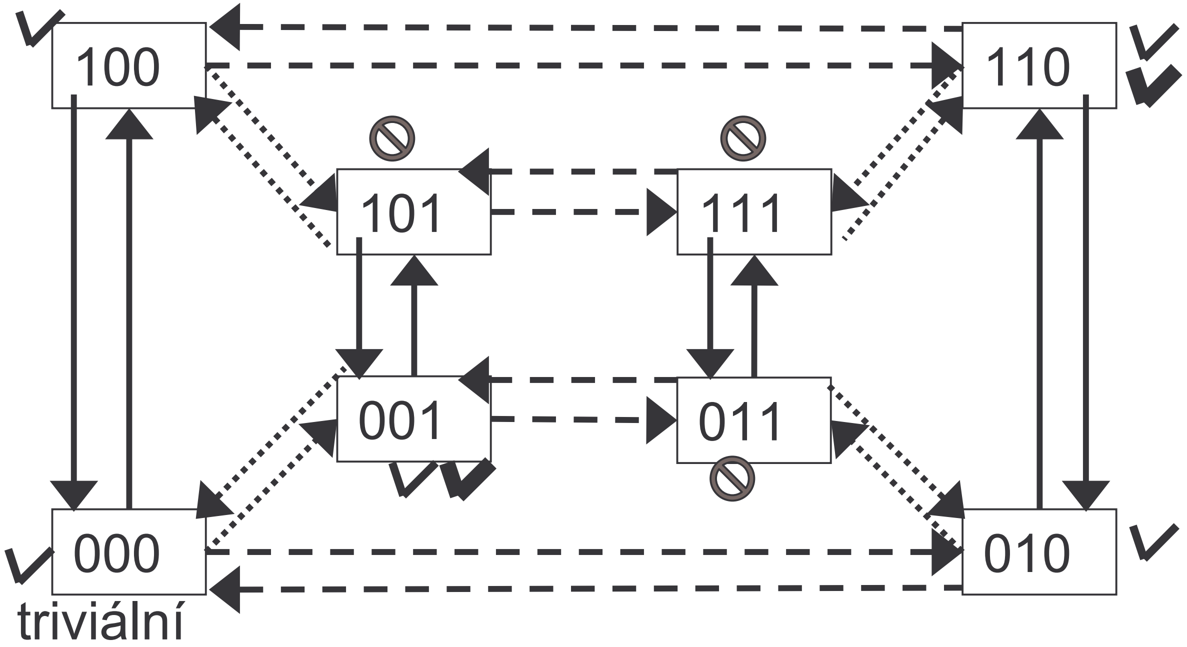
- Definice: stav
  Nechť $X = \{x_1, x_2, \ldots, x_n\}$ jsou konfigurační proměnné problému Π. Nechť $Z = \{z_1, z_2, \ldots, z_m\}$ jsou vnitřní proměnné algoritmu A řešícího instanci I problému Π. Pak každé ohodnocení s proměnných $X \cup Z$ je <u>stav</u> algoritmu A řešícího I.

- Definice: stavový prostor
  Nechť $S = \{s_i\}$ je množina všech stavů algoritmu A řešícího I. Nechť $Q = \{q_j\}$ je množina operací $S \to S$ takových, že $q_j(s_i) \neq s_i$ pro všechna s_i, q_j. Pak dvojici (S,Q) nazveme <u>stavovým prostorem</u> algoritmu A řešícího I.

Graf stavového prostoru

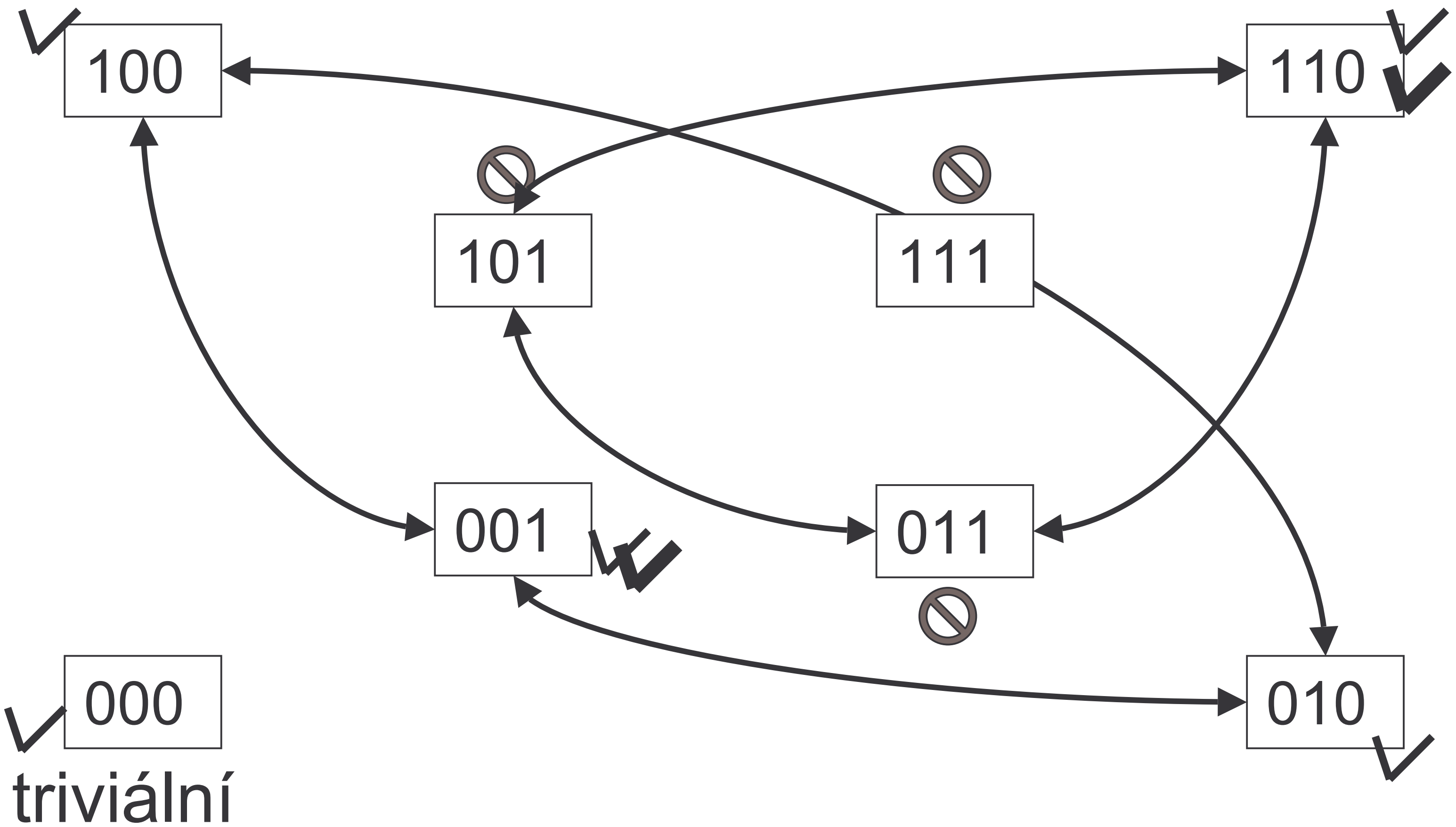
- Formální obrat, který dovolí přenést grafovou terminologii a algoritmy na stavový prostor
- Definice: tah
  Nechť $s \in S$ je (jeden konkrétní) stav a $q \in Q$ operace. Pak aplikace $q(s)$ se nazývá <u>tah.</u>
- Definice: graf stavového prostoru
  Nechť (S,Q) je stavový prostor algoritmu řešícího instanci problému. Pak orientovaný graf $H=(S, E)$, kde hrana $(e_i,e_j) \in E$ odpovídá tahu $s_j=q(s_i)$ pro $q \in Q$ se nazývá <u>grafem stavového prostoru algoritmu.</u>

Reverzní operace



- libovolnou vloženou věc lze vyjmout
- v grafu lze libovolně dlouho bloudit
- o následníku daného stavu nelze nic říci

Výměny



Stavový prostor je nesouvislý a proto bez doplnění dalšími operacemi neužitečný.

Problém obchodního cestujícího (TSP)

Dána množina n měst $C=\{c_1,c_2,\ldots,c_n\}$. Pro každá dvě města c_i,c_j je dána vzdálenost $d(c_i,c_j)>0$. Nalezněte uzavřenou túru, která prochází každým městem právě jednou a má nejmenší délku.

Dán graf $G=(V, E)$. Každá hrana (v_i,v_j) je ohodnocena vzdáleností $d(v_i,v_j)$. Nalezněte Hamiltonovu kružnici s minimálním součtem ohodnocení hran.

… optimalizační konstruktivní verze

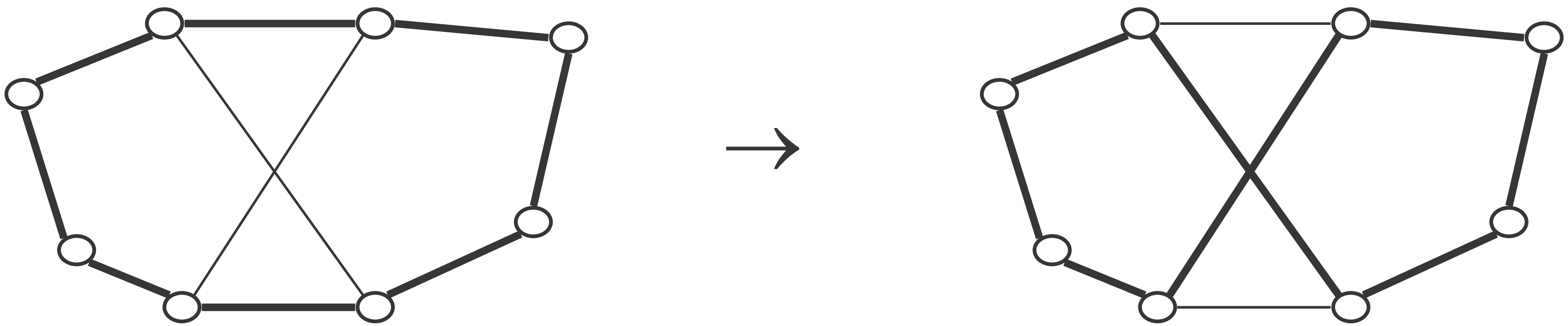
Varianty TSP

- Geometrický TSP:
  každý uzel ohodnocen souřadnicemi v rovině, každá hrana přímou vzdáleností (úplný graf)
- Metrický TSP:
  vzdálenosti splňují trojúhelníkovou nerovnost $|AC| \leq |AB| + |BC|$
- Obecný TSP:
  žádná podmínka pro vzdálenosti

Poznámka: hrana (v_i, v_j) neexistuje: vzdálenost $d(c_i, c_j)$ je nekonečná – formální nepříjemnost

Stavový prostor TSP

- Konfigurace:
  obecně podgraf grafu *G* (v závislosti na algoritmu kružnice, H. kružnice, cesta…)
- <u>Uzel stavového grafu = podgraf *G*</u>
- Operace:
  např. dvojzáměna na hranách



- <u>Hrana stavového grafu = dvojzáměna nad *G*</u>

Stav nebo cesta?

- Dosud probrané pojmy: úkolem je nalézt stav, který odpovídá (optimálnímu) řešení
- Některé úlohy (zejména UI): úkolem je nalézt cestu ke stavu (např. robotická manipulace scény)
- Stavový prostor je tedy množina posloupností akcí
- Což nebývá výhodné

Strategie pohybu stavovým prostorem

úplná, systematická

do hloubky, do šířky, nejlepší nejdříve

metoda větví a hranic (branch and bound)

nepřipadá Vám to povědomé?

Pohyb stavovým prostorem

- Aktuální stav, konfigurace příslušející aktuálnímu stavu
- Transformace aktuálního stavu pomocí operací ($\rightarrow$pohyb)
- Transformace nutno řídit $\rightarrow$ $\rightarrow$ strategie prohledávání
- Charakteristika algoritmu
  - stavový prostor
  - strategie prohledávání a ukončení
  - strategie prořezávání

Strategie

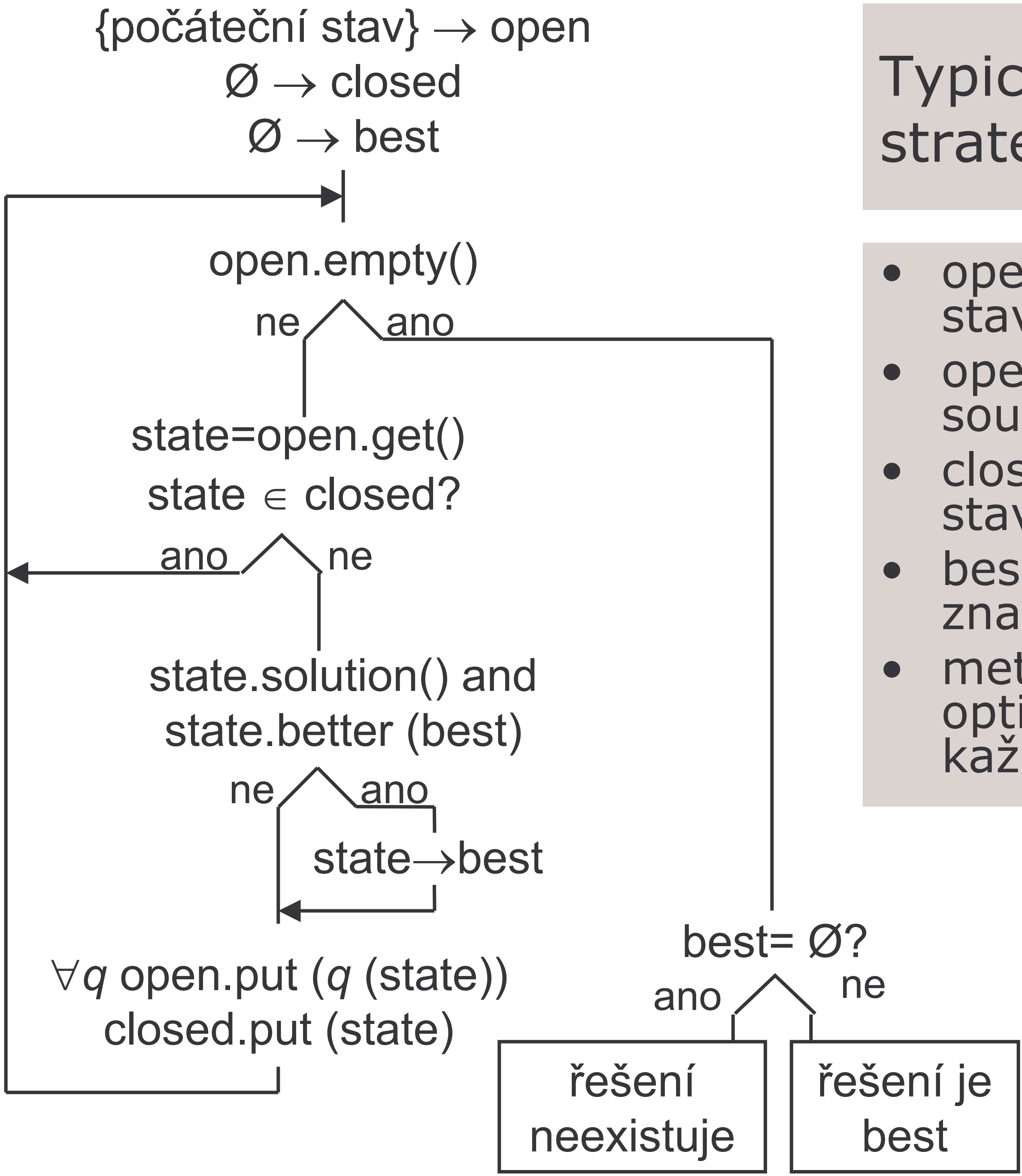
- Úplná strategie: navštívit všechny stavy kromě těch, o kterých víme, že nedávají (optimální) řešení

  Do not leave any stone unturned, unless you are sure there is nothing under it (Pearl)

- Systematická strategie: úplná, navštívit každý stav nejvýše jednou

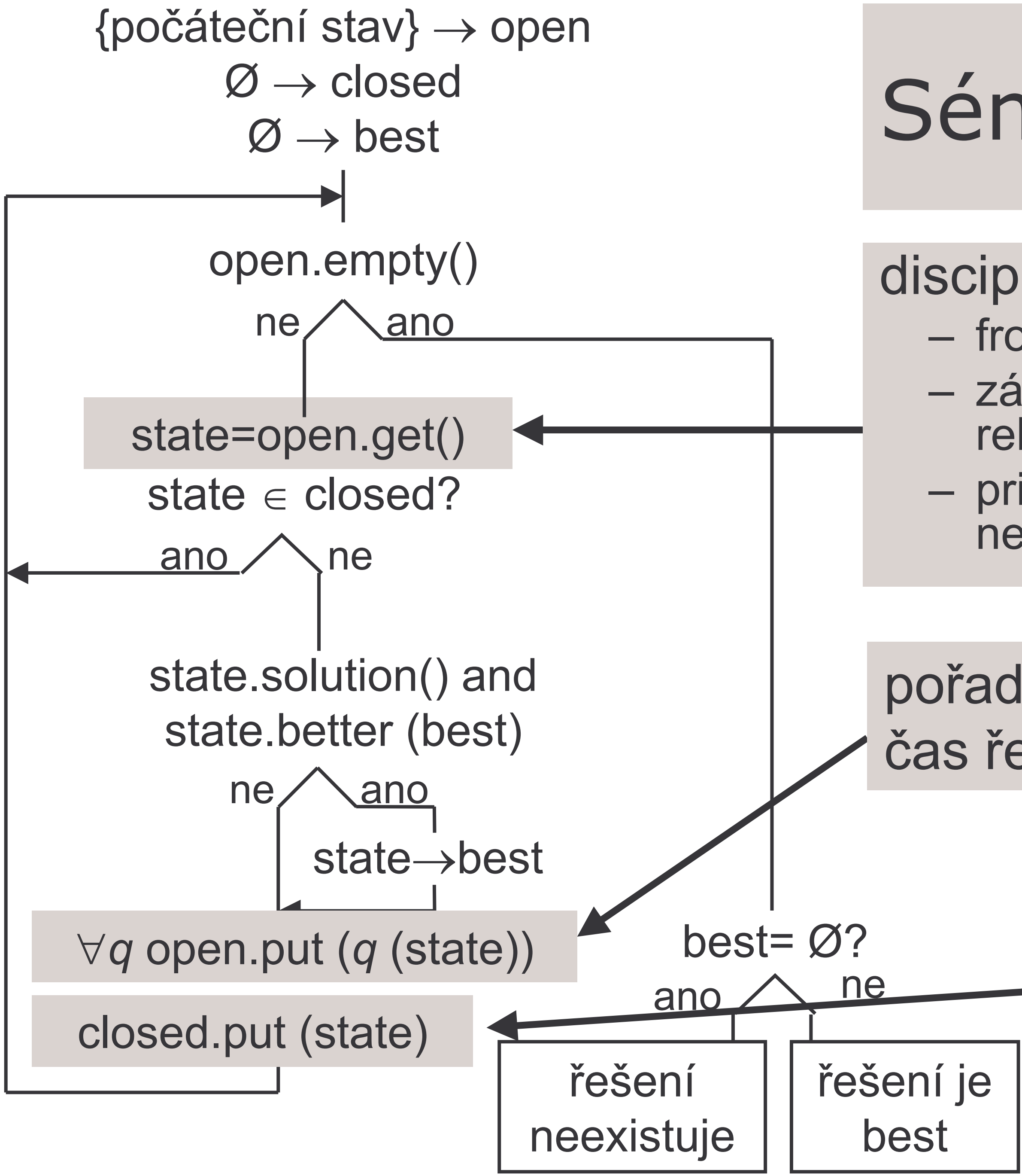
  Do not turn any stone more than once (Pearl)

Typická systematická strategie



- open, closed: množiny stavů
- open: neprozkomaní sousedé, o kterých víme
- closed: prozkoumané stavy
- best, state: stavy, Ø znamená „žádný stav“
- metoda better srovnává optimalizační kritéria; každý stav je lepší než Ø

Sémantika



disciplína struktury open:

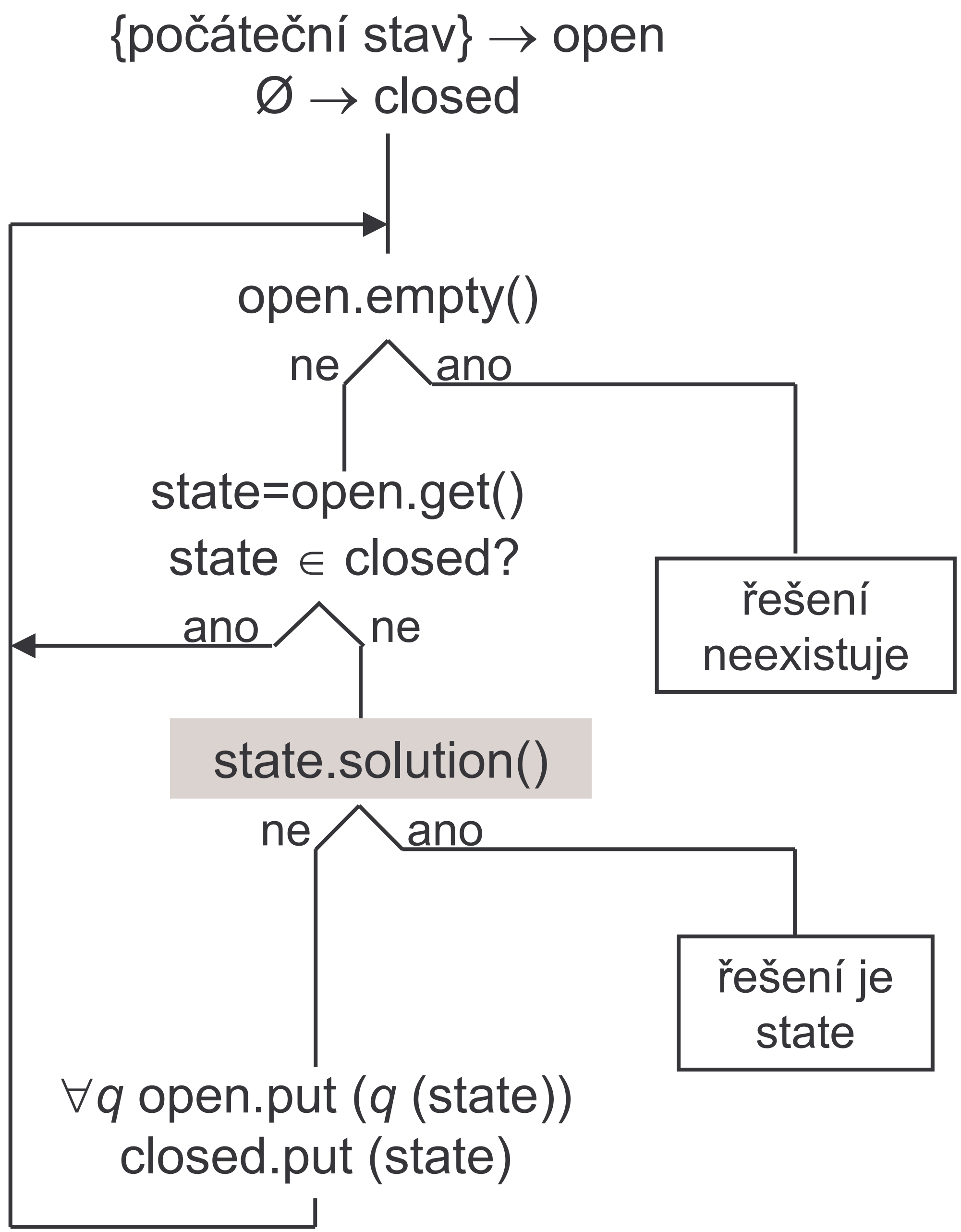
- fronta: do šířky
- zásobník: do hloubky (jako rekurzivní formulace)
- prioritní fronta: nejlepší nejdříve

pořadí ovlivní průměrný čas řešení

graf stavového prostoru je strom → není třeba closed

Rozhodovací a konstruktivní problémy

do prioritní fronty se stavy zařazují podle heuristické funkce, která strategii „navádí“ k řešení



Vlastnosti systematických strategií

- Nejhorší případ roven hrubé síle
- V případě neexistujícího řešení tento případ nastane
- Naleznou řešení, existuje-li
- Naleznou optimální řešení

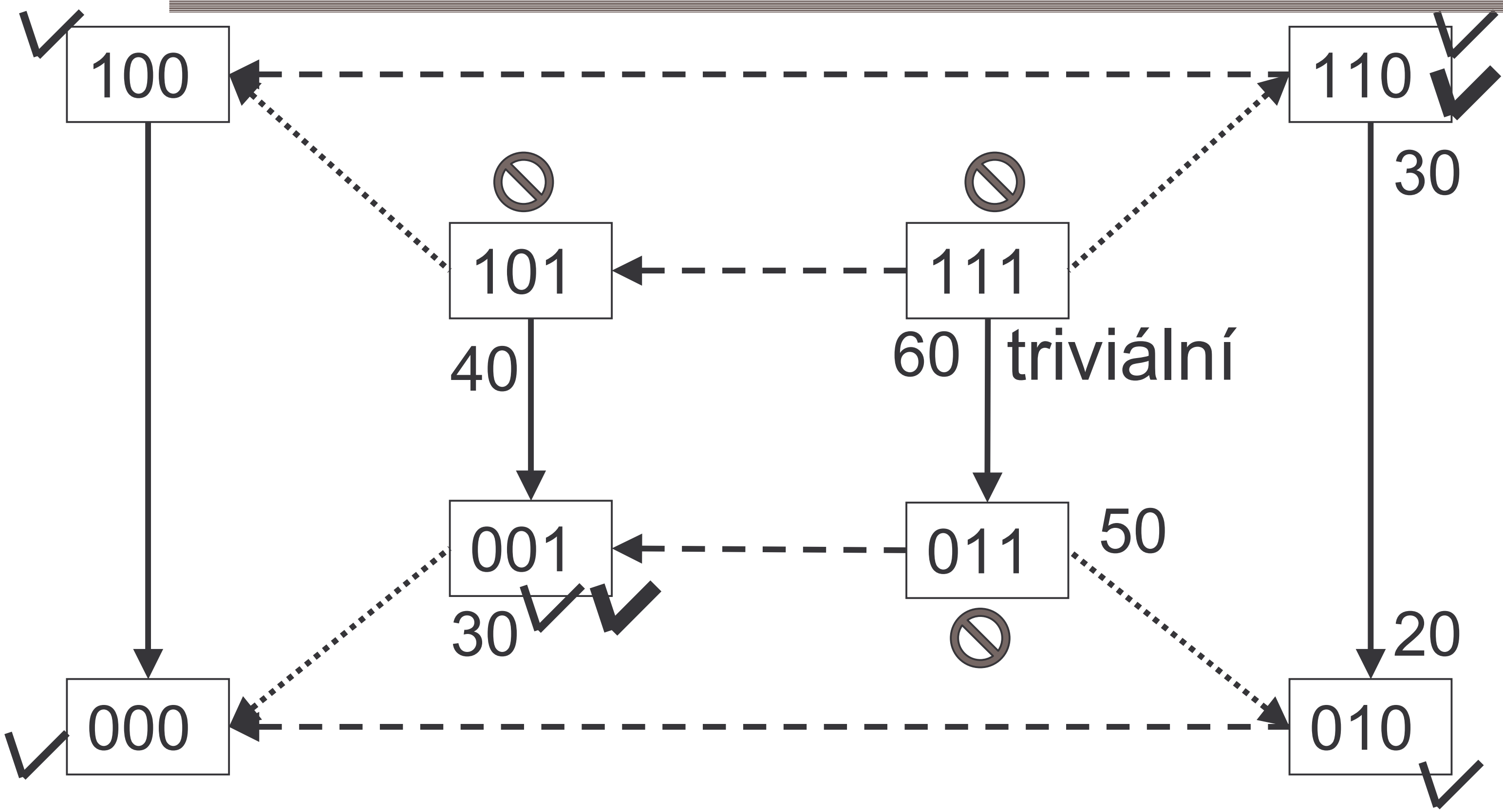
Vztah k A* a nejkratší cestě

- „Hledáme stavovým prostorem nejkratší cestu k řešení“
- Velikost grafu stavového prostoru u „neúnosných“ problémů rychle roste s velikostí instance
- A* předpokládá, že cena v každém uzlu závisí na tom, jak jsme se tam dostali → korekce ceny (label correcting)
- Cena uzlu ve stavovém prostoru záleží jen na konfiguraci (uzlu) → není třeba korekce (label setting)

Prořezávání

ve stavu *s* není řešení a cena <30, stop této větvi …proč?

111: není řešení, cena 60
110: je řešení, cena 30, stop této větvi, návrat
011: není řešení, cena 50>30, pokračuj
010: je řešení, cena 20<30, stop této větvi, návrat

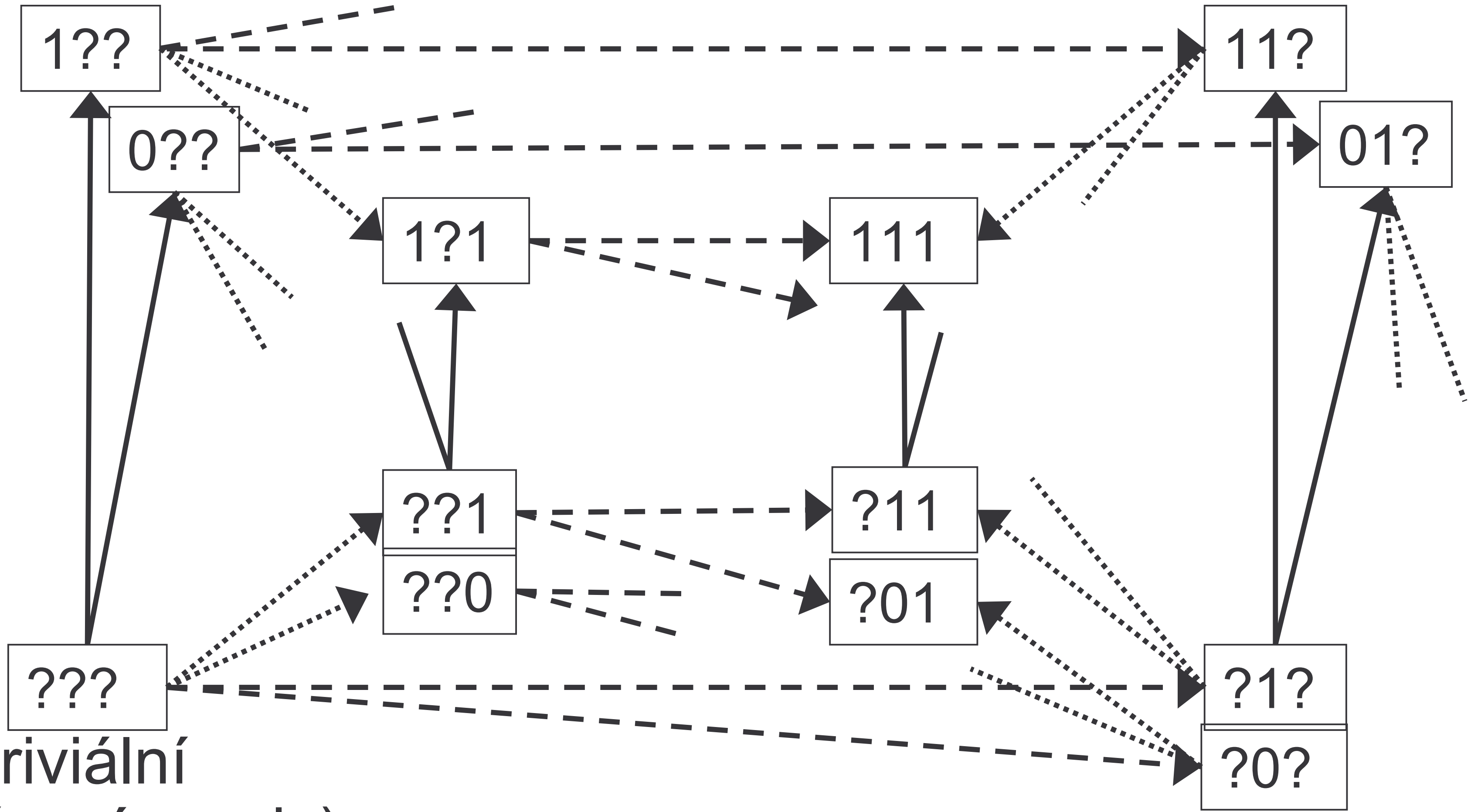


Prořezávání

- Na základě
  - dosud dosažené hodnoty optimalizačního kritéria
  - splnění omezujících podmínek
  - znalostí optimalizačního kritéria a omezujících podmínek
- můžeme vyloučit (prořezat) určité oblasti stavového prostoru (větve hledání)
- Odhady dolní/horní meze optimalizačního kritéria mohou pocházet i z jiných metod

Prostor prohledávání
(search space)

Postupné ohodnocování proměnných



triviální
(nevíme nic)

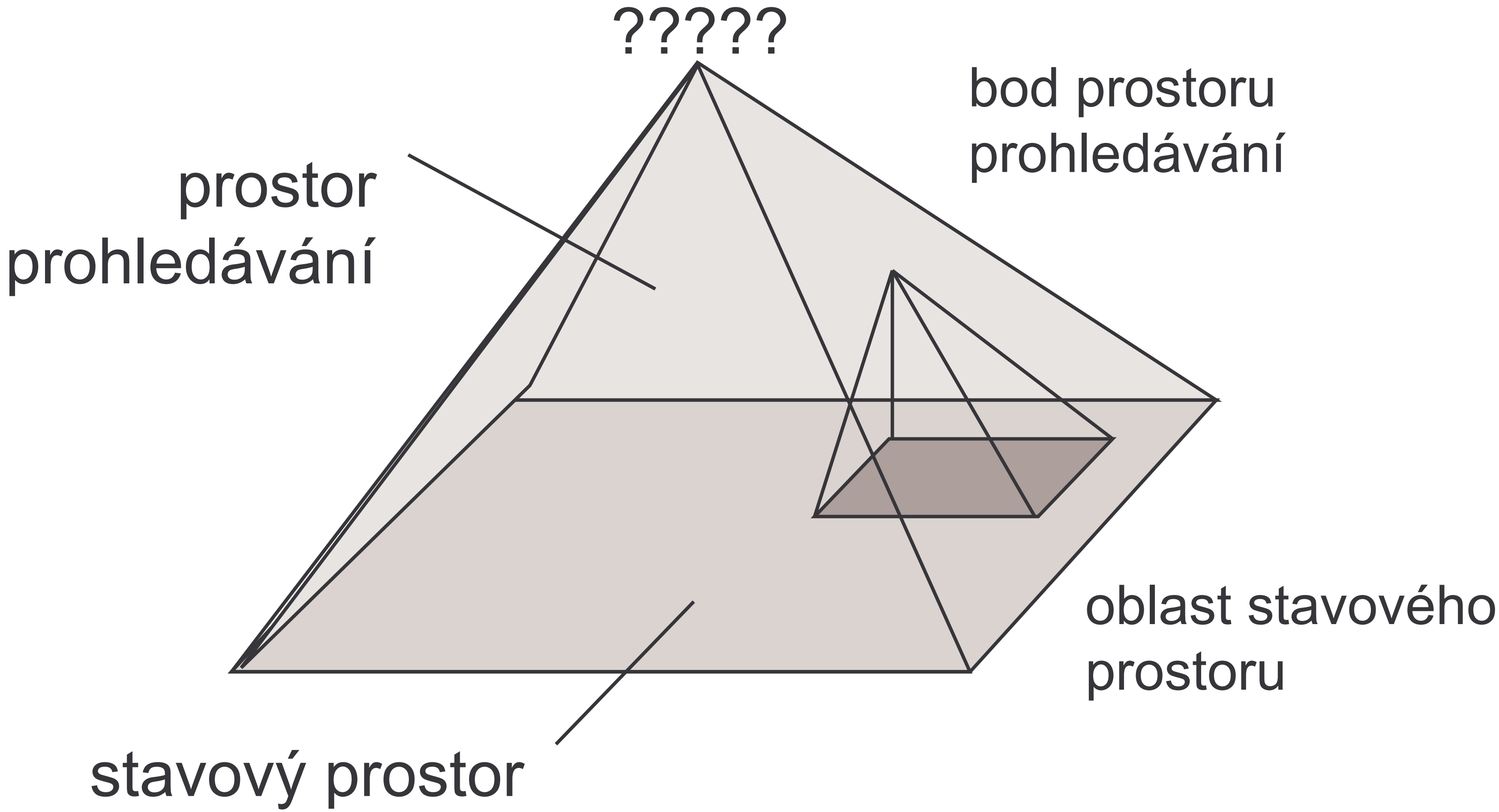
Formalizace

- ke každé konfigurační proměnné x_i přiřadíme pomocnou proměnnou z_i s významem „x_i má platnou hodnotu“

  nebo

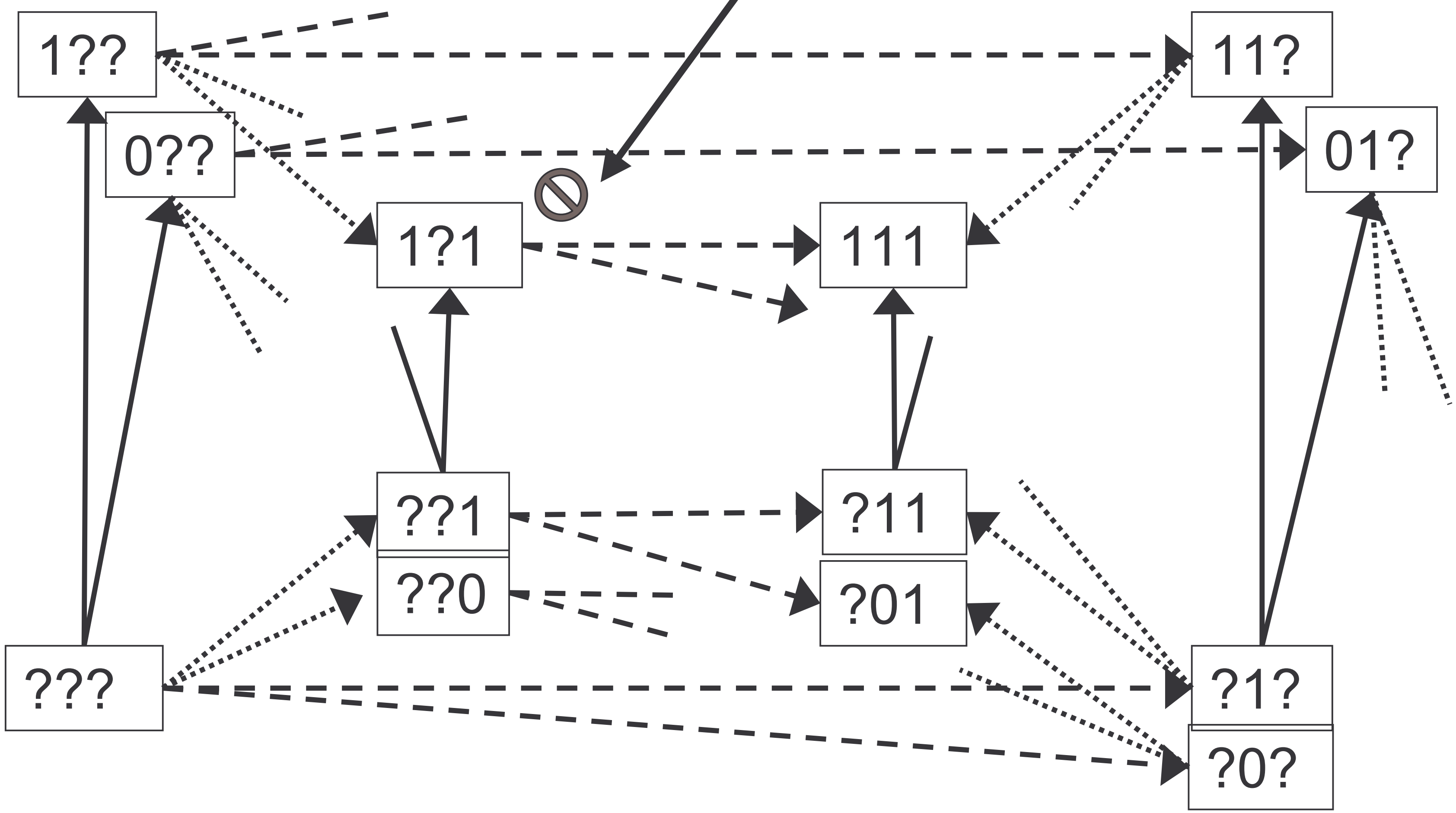
- rozlišíme stavový prostor, kde každému prvku přísluší jedna konfigurace, a prohledávací prostor, kde každému prvku přísluší množina konfigurací

Vztah ke stavovému prostoru



Prořezávání

vztahuje se k *oblasti* stavového prostoru



Pohyb v prohledávacím prostoru

- Typické strategie nebývají úplné, natož systematické
- Typický krok prohledávání:
  - vyber proměnnou
  - vyber hodnotu proměnné
- Možnost odvolat nastavení proměnné (backtracking)
- Podrobněji viz lokální heuristiky

Problém diskrétního rozmístění

- Dáno:
  - množina n modulů $K=\{k_1, \dots, k_n\}$
  - množina m pozic $P=\{p_1, \dots, p_m\}$, $m \geq n$
  - propojení modulů jako hypergraf G (K, N), kde N je množina spojů n
  - cenová funkce W (R, n), která pro každé přiřazení R: $K \rightarrow P$ odhadne cenu realizace spoje n.
- Nalézt:
  - prosté přiřazení R: $K \rightarrow P$ (rozmístění), které minimalizuje součet ohodnocení W (R, n) přes všechny spoje.

Strategie

- Poznámka:
  zobrazení R můžeme chápat jako
  - jednu proměnnou, s triviální hodnotou Ø, kterou měníme, až splní omezení (prosté zobrazení)
  - pro každý modul $k_i \in K$ jednu proměnnou r_i
- Strategie:
  - na začátku není rozmístěno nic
  - vybereme modul
  - vybereme pro něj nejlepší pozici
  - opakujeme, dokud nejsou rozmístěny všechny moduly