

"Agent Hledač"

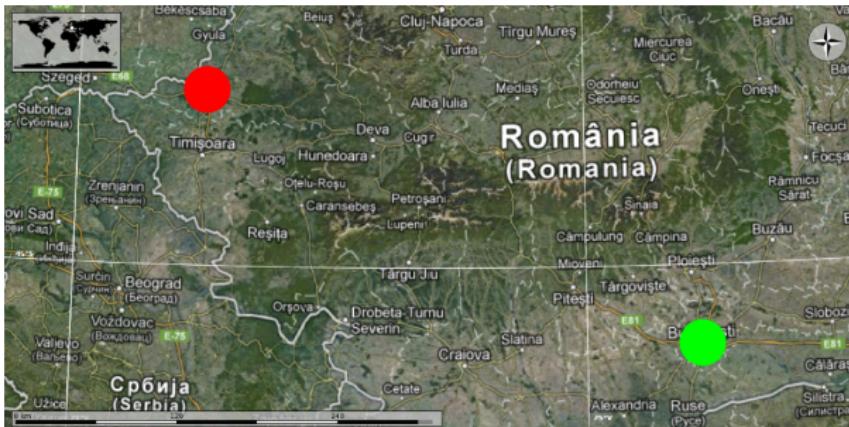
(3. přednáška)

Přehled 3. přednášky

- v této přednášce se budeme zabývat "goal-based" agenty
- připomeňme, že "goal-based" agent se snaží nalézt posloupnost akcí, vedoucích k cíli
- budeme zkoumat obecné "postupy", jak tuto posloupnost najít

Příklad — dovolená v Rumunsku

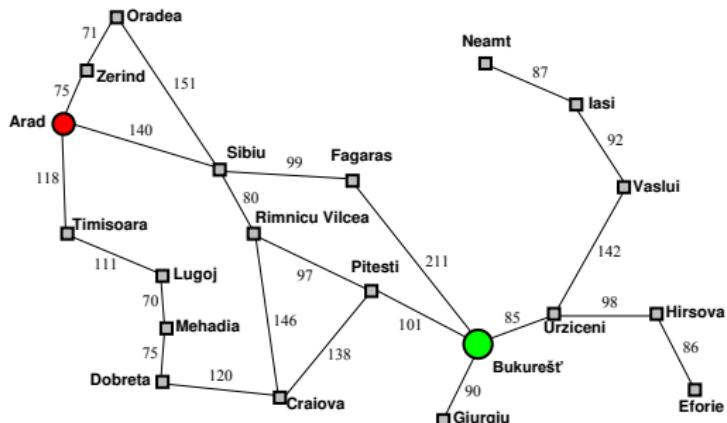
Agent je na prázdninách v Rumunsku. Momentálně se nachází v Aradu, zítra mu letí letadlo z Bukurešťe.



Příklad — dovolená v Rumunsku

Agent je na prázdninách v Rumunsku. Momentálně se nachází v Aradu, zítra mu letí letadlo z Bukurešť.

Pojďme trochu abstrahovat.



Dovolená v Rumunsku (cesta v grafu)

Přímo se nabízí grafová formulace

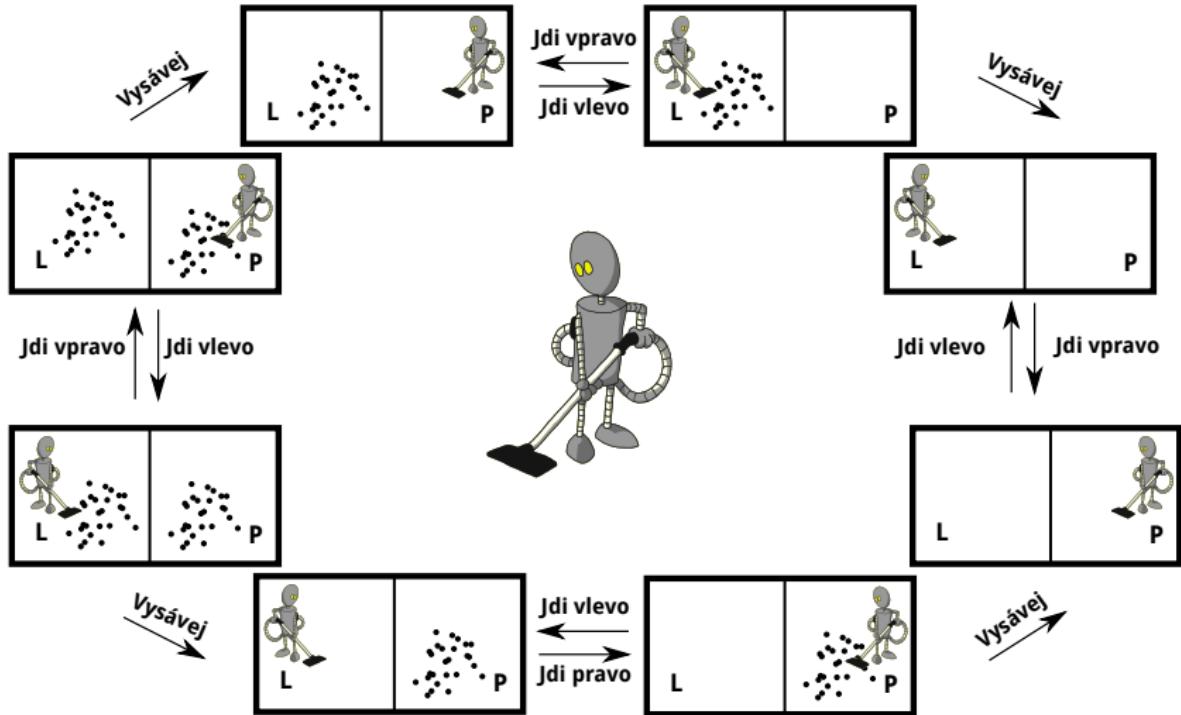
- prostředí je dánou grafem (silniční síť Rumunska)
- agent se pohybuje po hranách tohoto grafu mezi vrcholy
- jeho cílem je najít cestu z vrcholu Arad do vrcholu Bukurešť
- (případně je cílem najít **nejkratší** cestu)

Agent vysavač — hledání pořádku

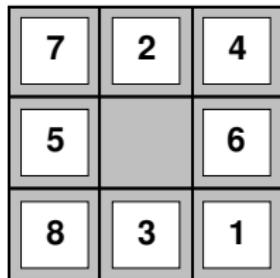
Robotický vysavač, nacházející se (pro jednoduchost) v místnosti rozdělené na dvě pole L a P. Umí pohybovat **doprava**, **doleva**, a **vysávat**. Cílem je dosáhnout čisté místnosti.

- vrchol grafu — stav světa + pozice robota
- hrana grafu — odpovídá jedné ze tří možných akcí
- cílový vrchol — stav, kde jsou obě pole čistá

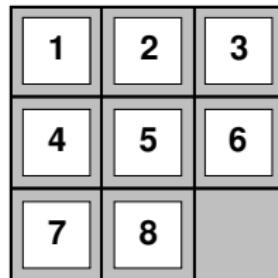
Agent vysavač — hledání pořádku



Agent 008 — Loydova osmička



Počáteční stav



Cílový stav

- vrchol grafu — pozice kostiček
- hrana grafu — pohnutí kostičkou (volným místem)
- cílový vrchol — kostičky jsou srovnанé podle čísel

Optimální řešení (t.j. na nejmenší počet tahů) n -verze tohoto hlavolamu je NP-těžký problém.

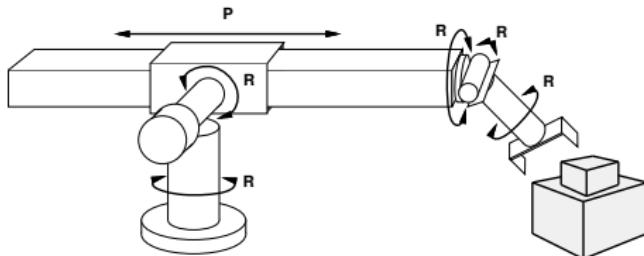
Obchodní cestující (TSP)

Obchodní cestující chce navštívit nějaká města a vrátit se na začátek. V jakém pořadí je má navštěvovat tak, aby minimalizoval vzdálenost, kterou bude muset urazit?

- vrchol grafu — kombinace následujících údajů: město, ve kterém se obchodní cestující nachází a stav navštívenosti jednotlivých měst
- hrana grafu — silnice spojující města
- cílový vrchol — vrchol odpovídající počátečnímu městu, ve kterém jsou všechna města navštívena

NP-těžký problém.

Montážní linka



- vrchol grafu — kombinace poloh jednotlivých kloubů robotického ramene a jednotlivých součástí
- hrana grafu — pohnutí jednotlivého robotického kloubu
- cílový vrchol — stav, kde jednotlivé součásti jsou složeny

Hledáme řešení — strom akcí

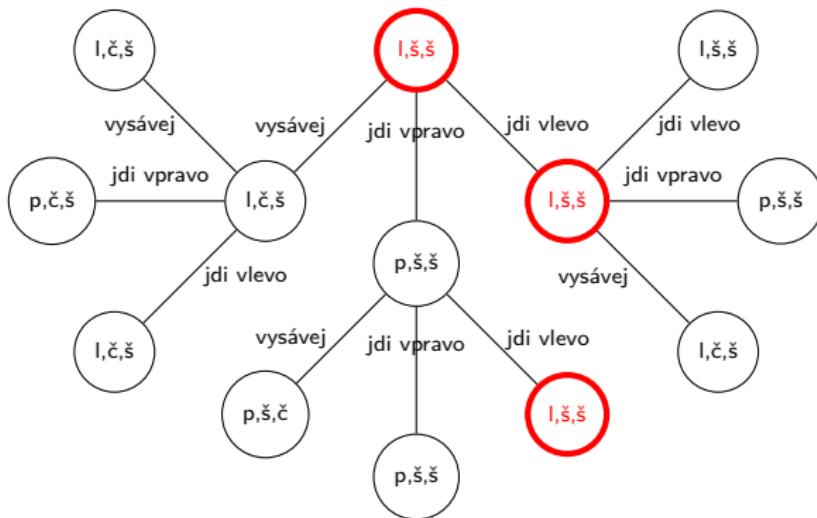
Idea

Procházet všechny možné posloupnosti akcí a zjistit, která vede k cíli

Postupným procházením vytváříme strom

- uzly ve stromu jsou posloupnosti akcí
- každému uzlu odpovídá stav světa (po provedení dané posloupnosti akcí)
dvěma různým uzelům mohou odpovídat stejné stavy světa (!)
- chceme najít uzel, jehož stav je cílovým stavem

Strom akcí — robot vysavač



○

různé uzly odpovídající stejným stavům

Prohledávání stromu akcí — idea

Postupně budujeme strom akcí, dokud nenarazíme na cílový stav.

- v každém kroku dle nějaké strategie vybereme uzel ve stromu akcí
- v tomto uzlu strom rozšíříme
 - za každou akci, kterou je možné v daném uzlu (resp. stavu) provést přidáme do stromu akcí nový uzel

Prohledávání stromů — tree search

```
def treeSearch( problem, strategy ):
    # Inicializuj strom.
    tree = node(problem.initial_state())
    while True:
        # Zvol kandidata k expanzi dle strategie.
        leaf_node = strategy(problem, tree)
        # Mame reseni, vrat posloupnost kroku.
        if problem.is_goal( leaf_node.state() ):
            return leaf_node
        # Expanduj kandidata (pridej do stromu
        # uzly, ktere sousedi s kandidatem).
        stav = leaf_node.state()
        for a in stav.possible_actions():
            cil_stav = stav.act(a)
            child = node(cil_stav)
            leaf_node.add_child( a, child )
```

Prohledávací strategie — hodnotící kritéria

- úplnost (vždy nalezne cílový stav, pokud existuje)
- optimálnost (posloupnost akcí vedoucí do cílového stavu bude optimální)
- časová složitost (kolik uzlů je třeba vygenerovat)
- paměťová náročnost (kolik uzlů je třeba držet v paměti)

Časová složitost a paměťová náročnost se měří jako funkce

b — faktor větvení (branching factor)

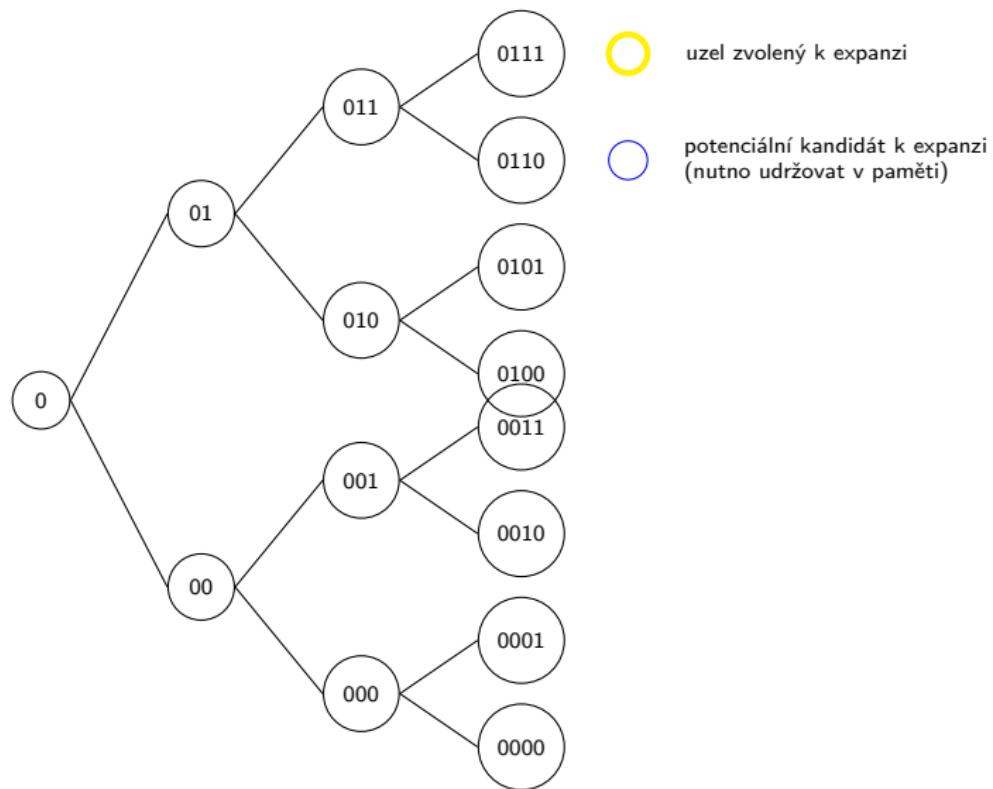
d — hloubka optimálního řešení

m — hloubka prohledávaného stromu

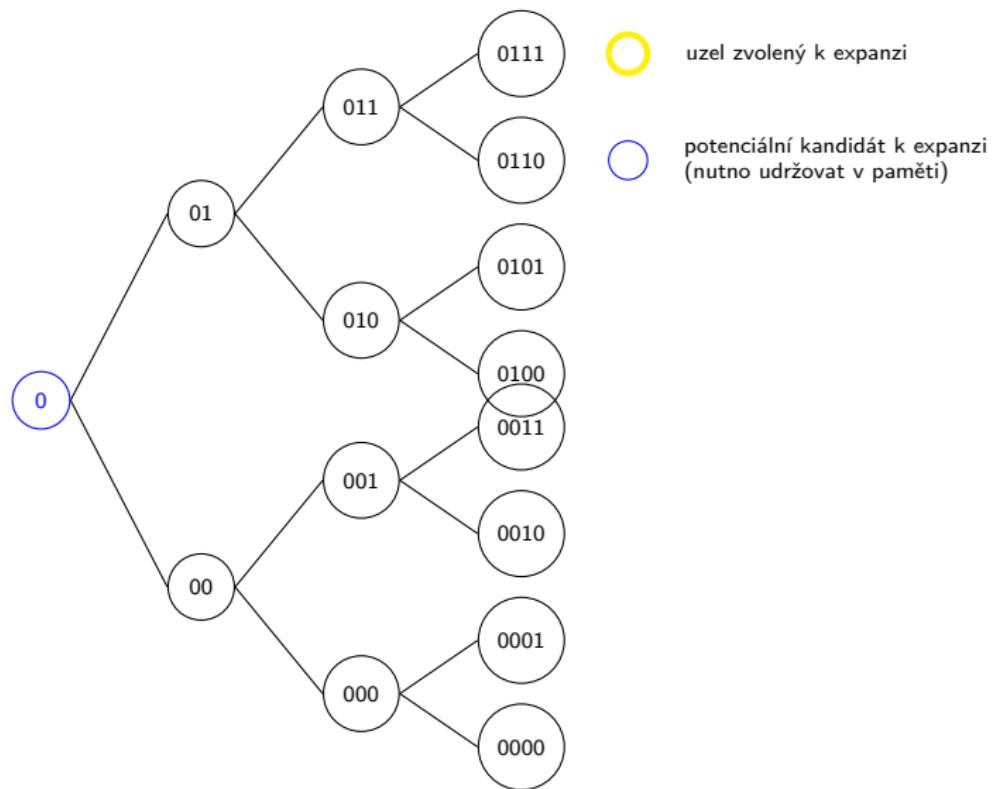
Prohledávací strategie — uninformed search

- prohledávání do šířky (BFS)
- prohledávání do hloubky (DFS)
- omezená hloubka (depth limited, DLS)
- iterování hloubky (iterative deepening, IDS)

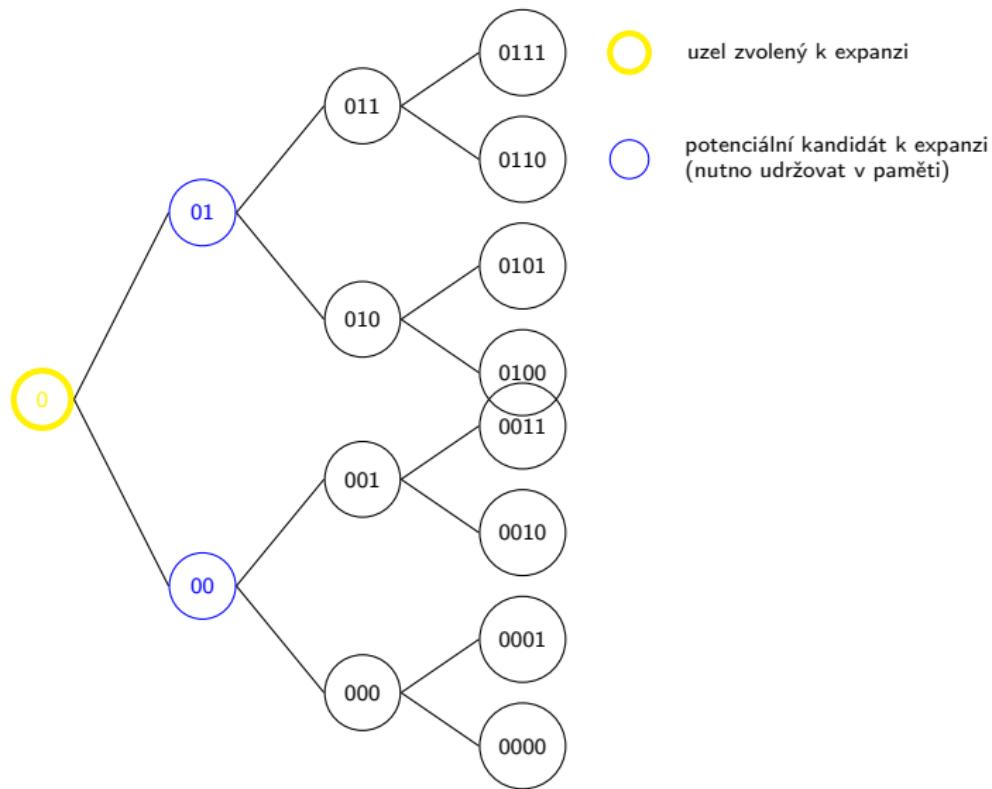
BFS — prohledávání do šířky



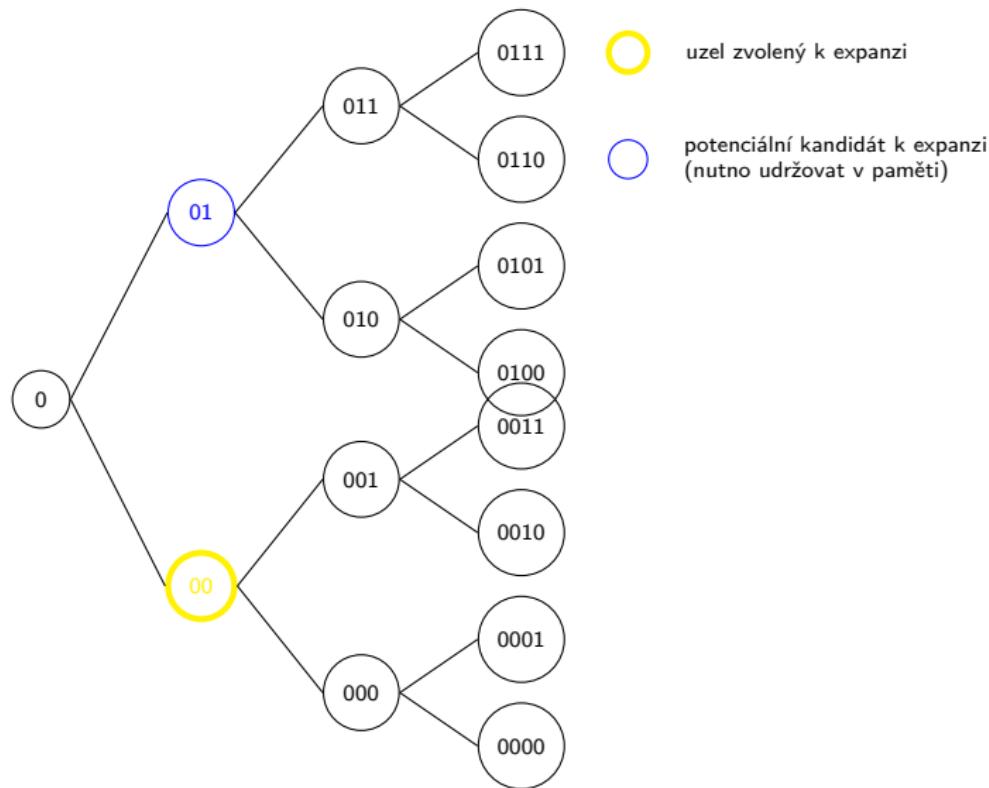
BFS — prohledávání do šířky



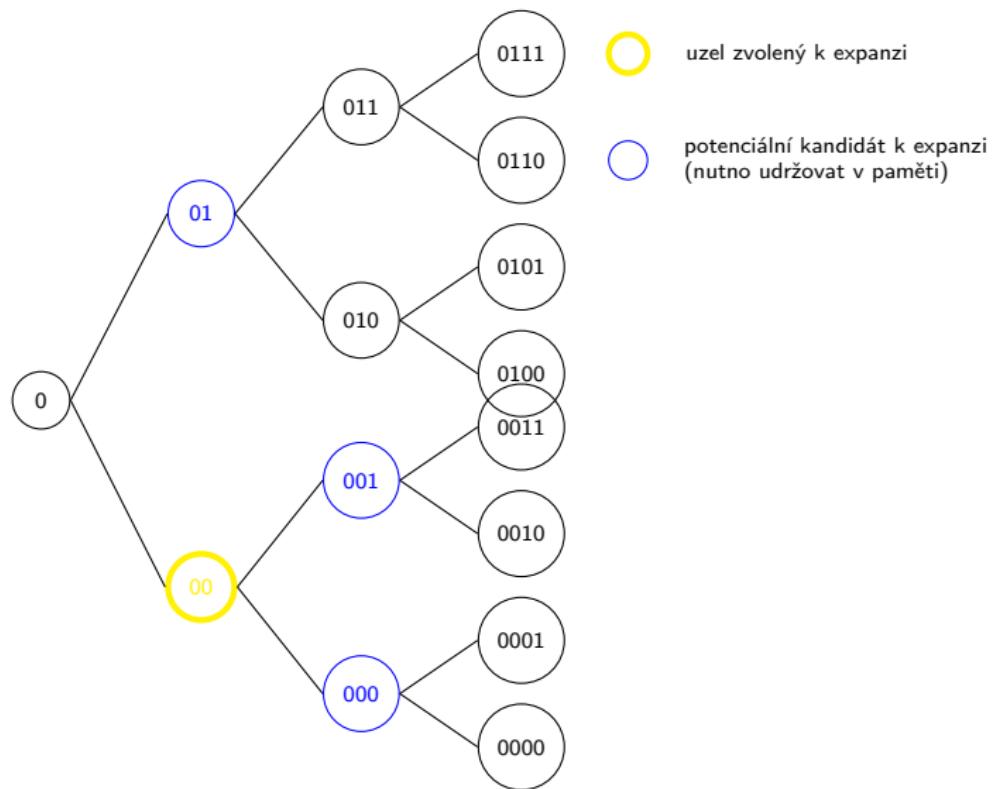
BFS — prohledávání do šířky



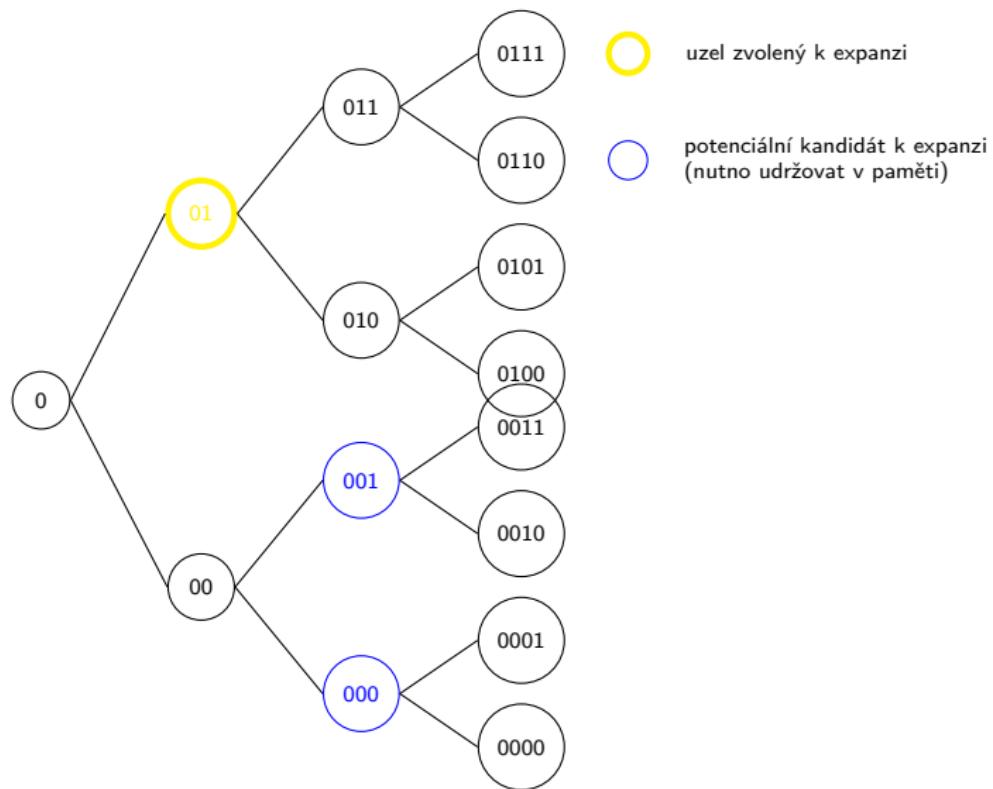
BFS — prohledávání do šířky



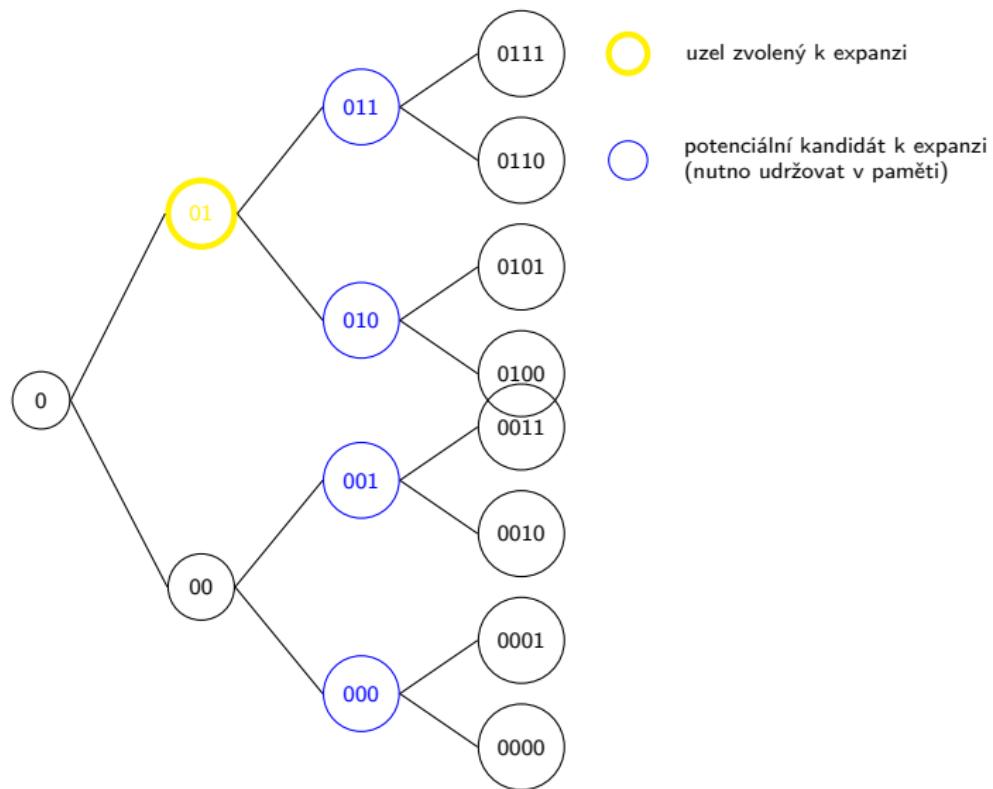
BFS — prohledávání do šířky



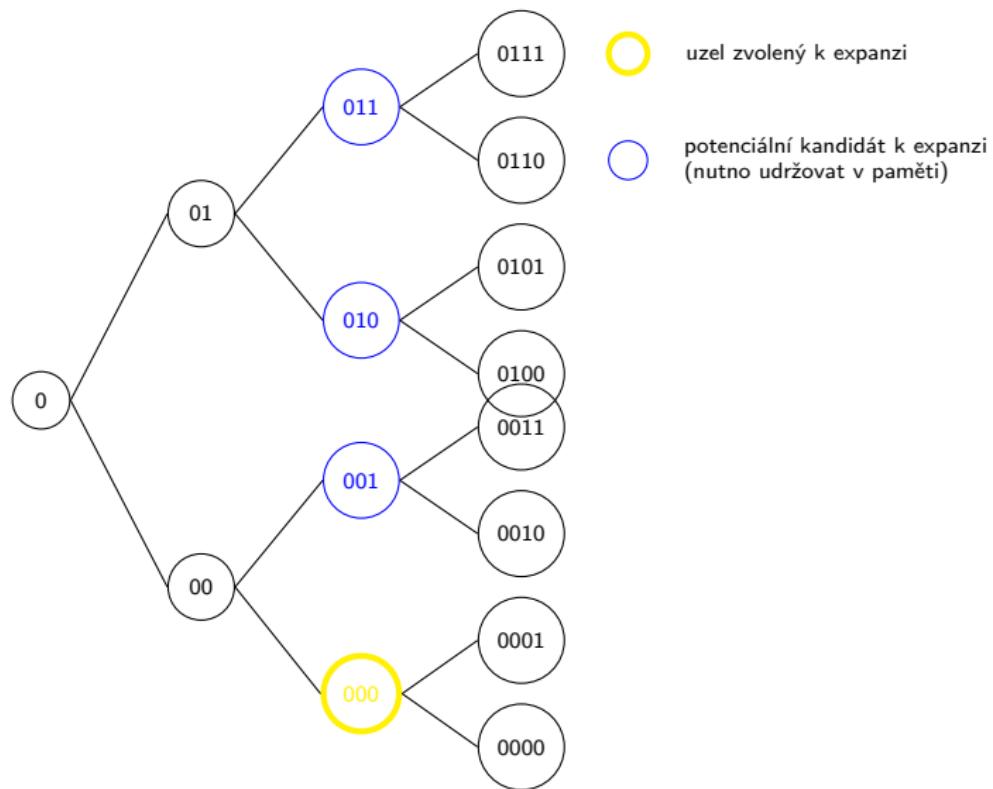
BFS — prohledávání do šířky



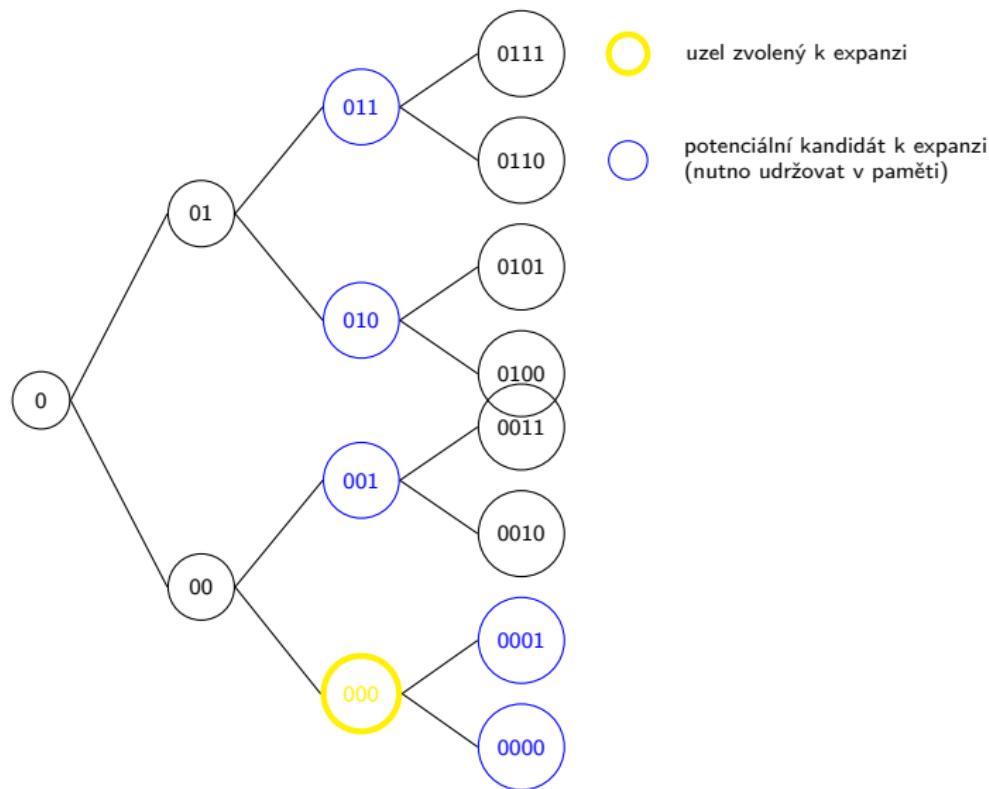
BFS — prohledávání do šířky



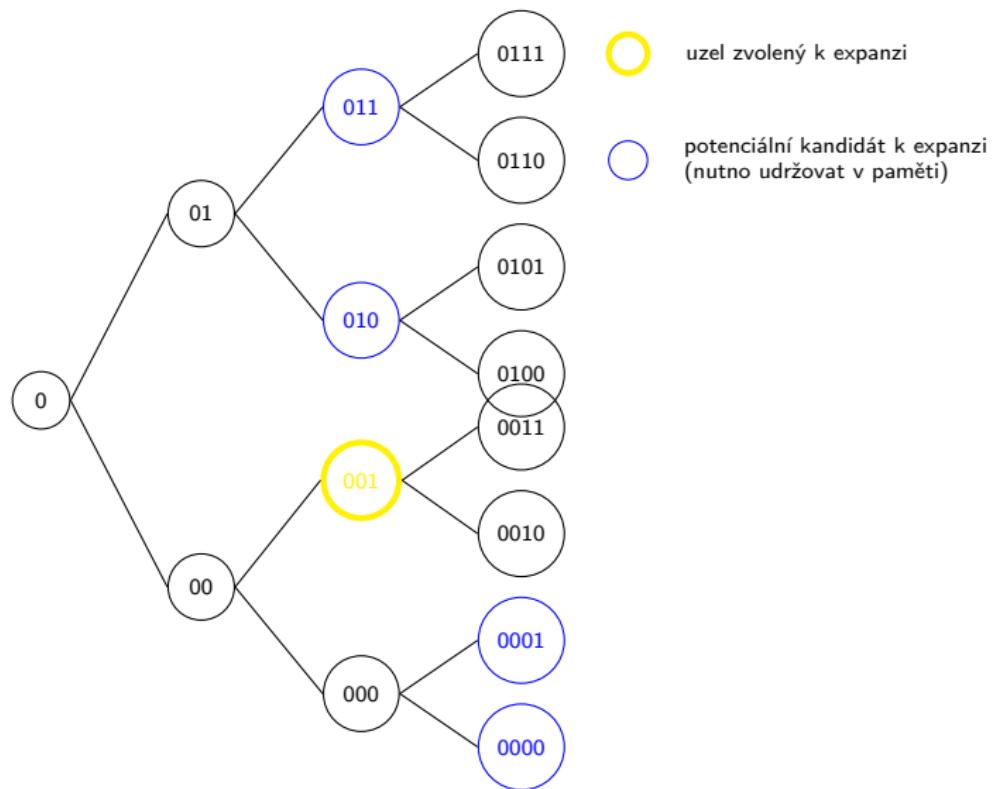
BFS — prohledávání do šířky



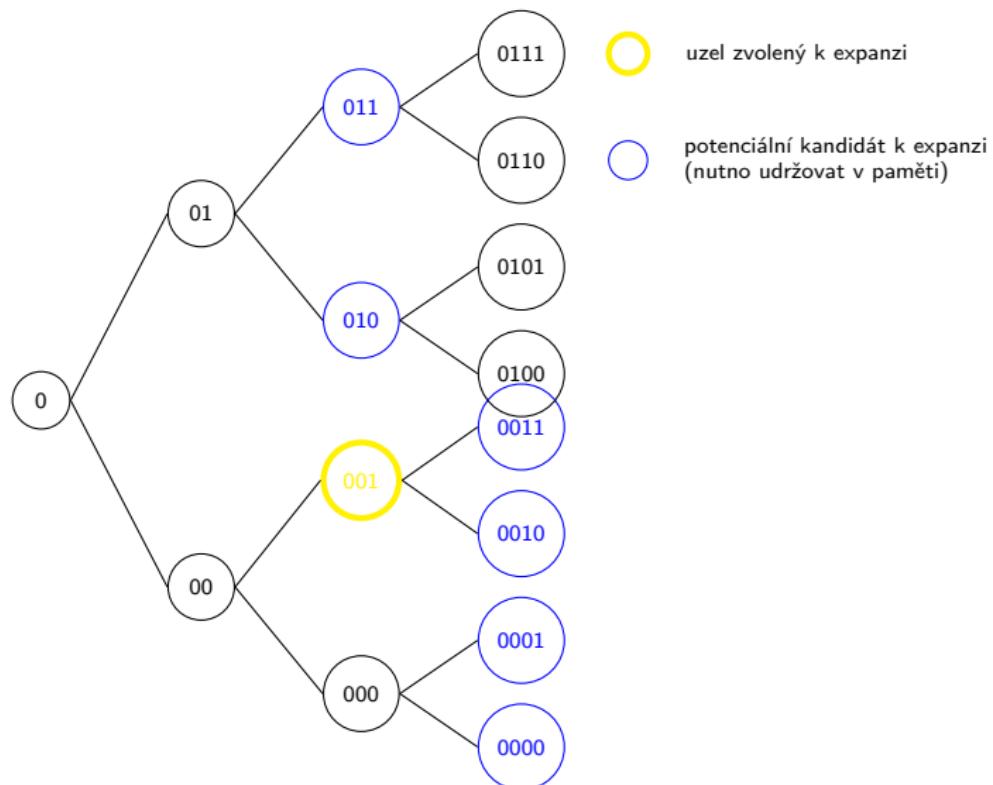
BFS — prohledávání do šířky



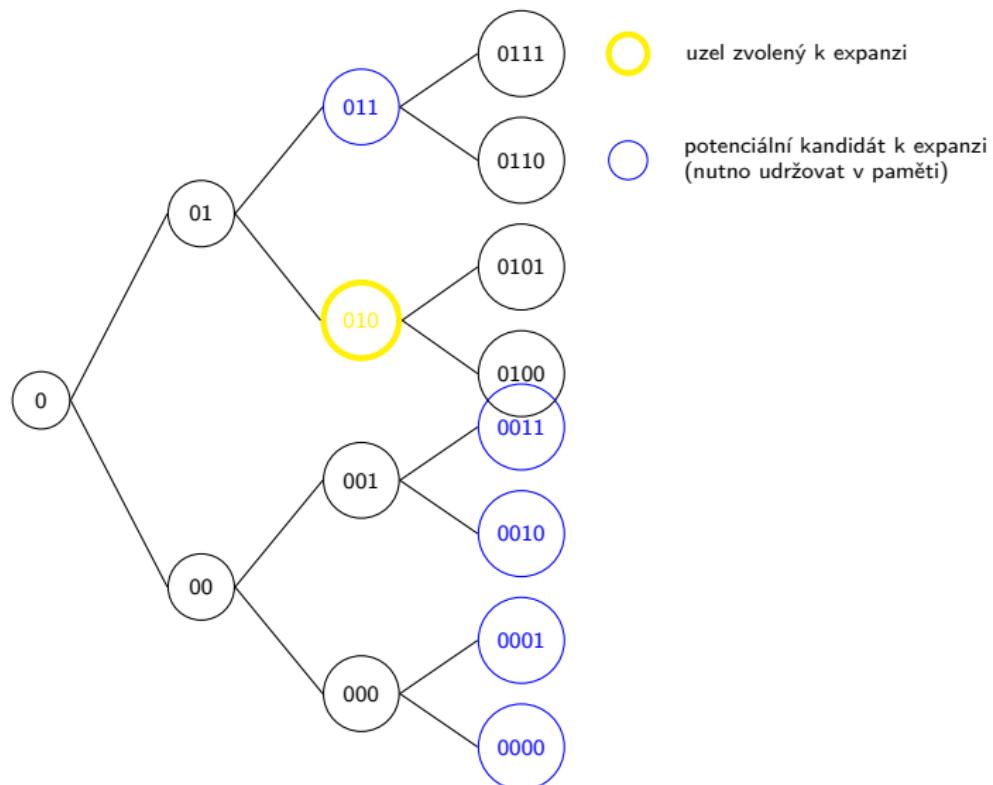
BFS — prohledávání do šířky



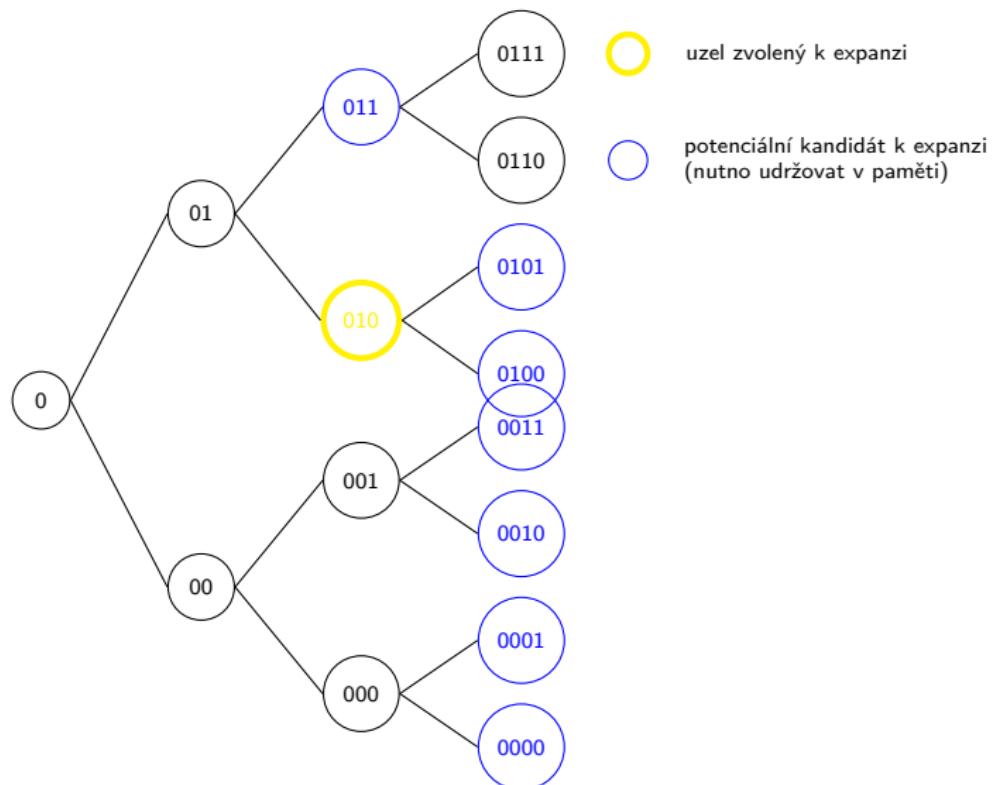
BFS — prohledávání do šířky



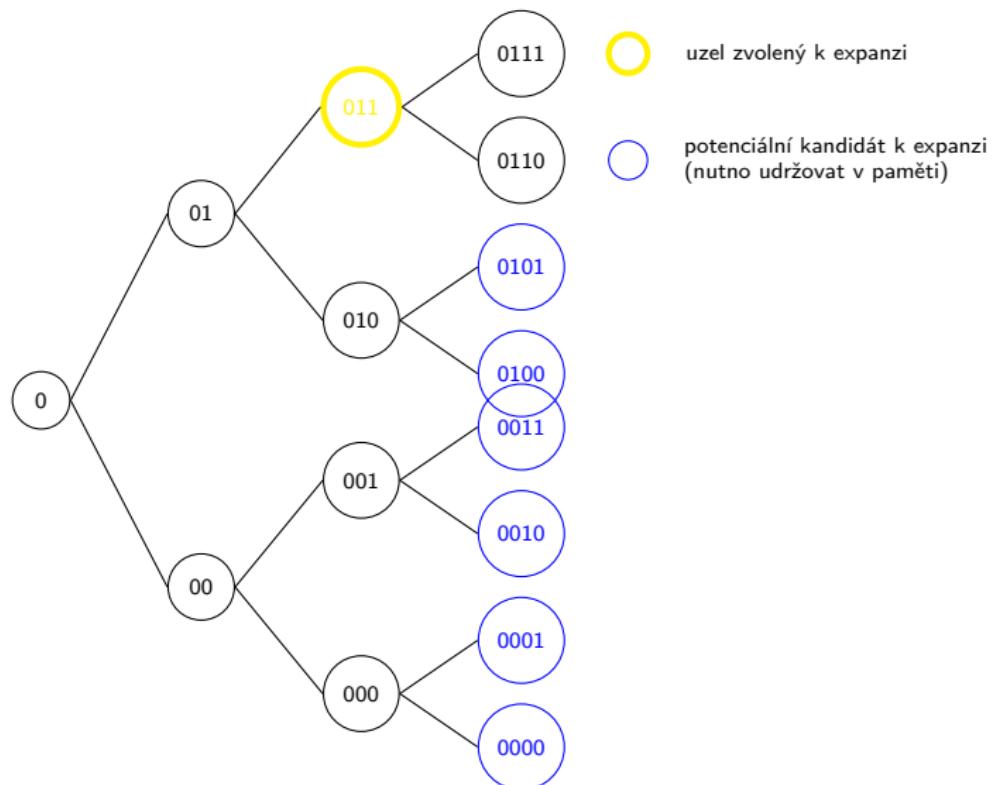
BFS — prohledávání do šířky



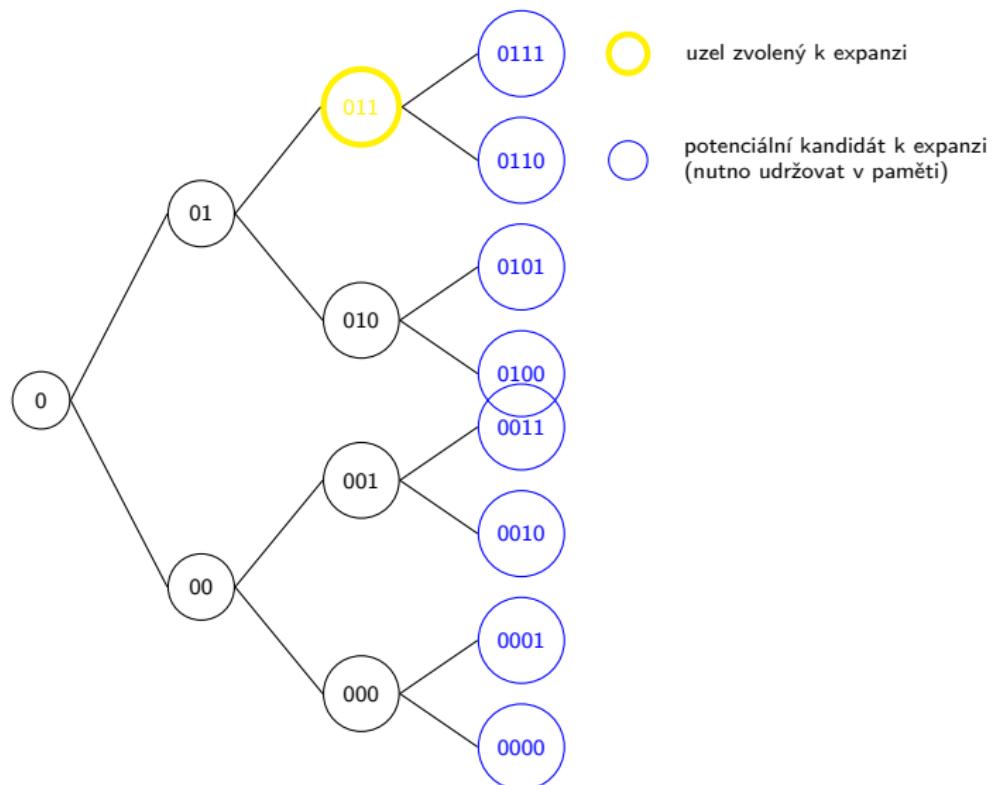
BFS — prohledávání do šířky



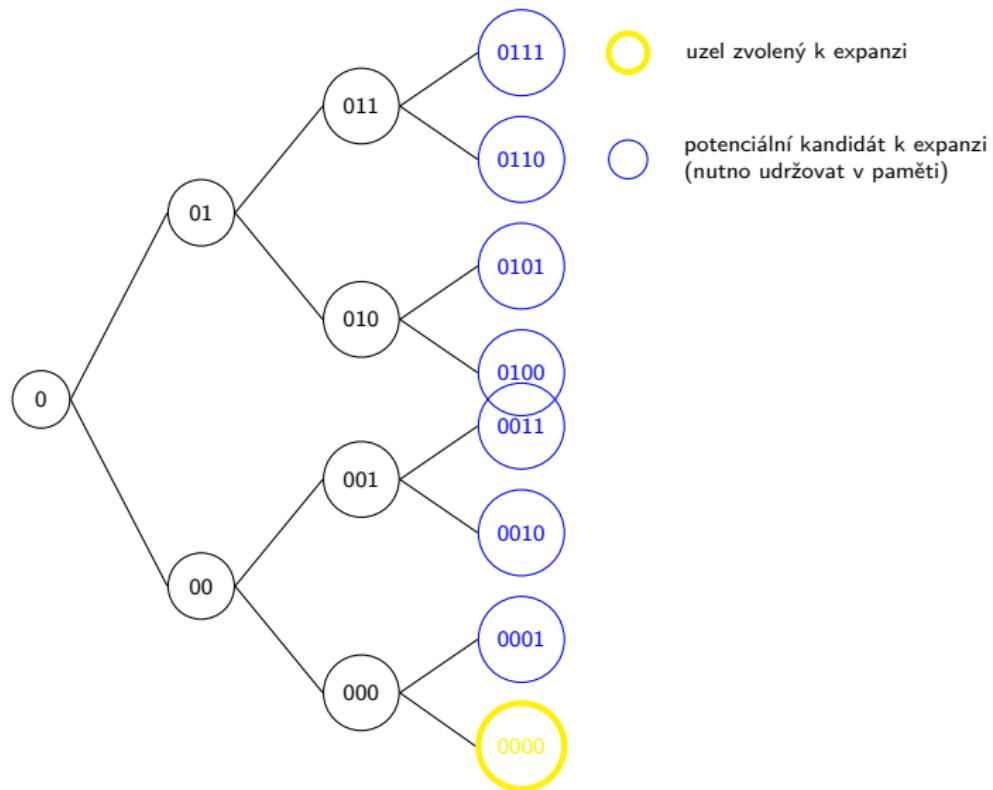
BFS — prohledávání do šířky



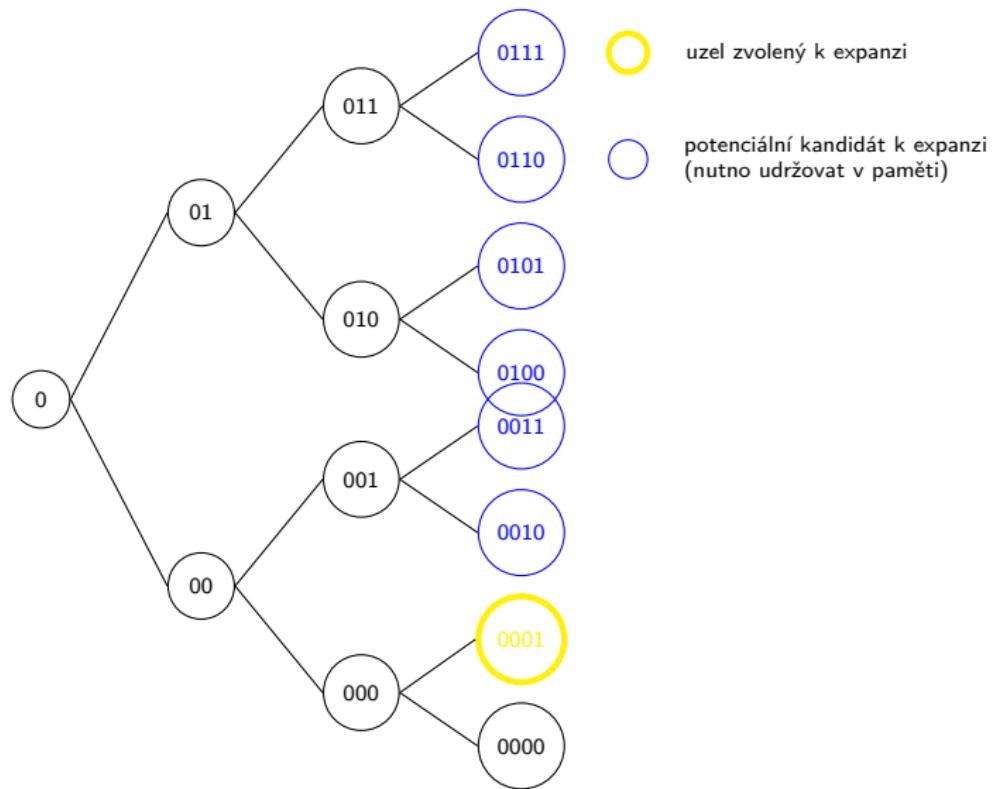
BFS — prohledávání do šířky



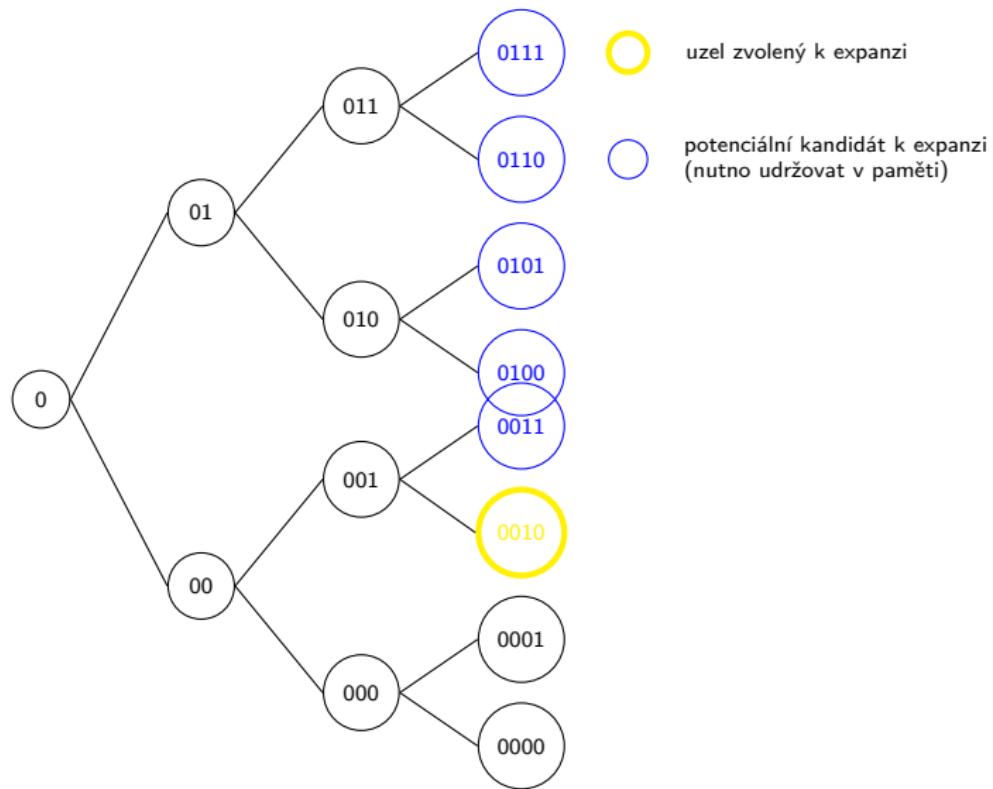
BFS — prohledávání do šířky



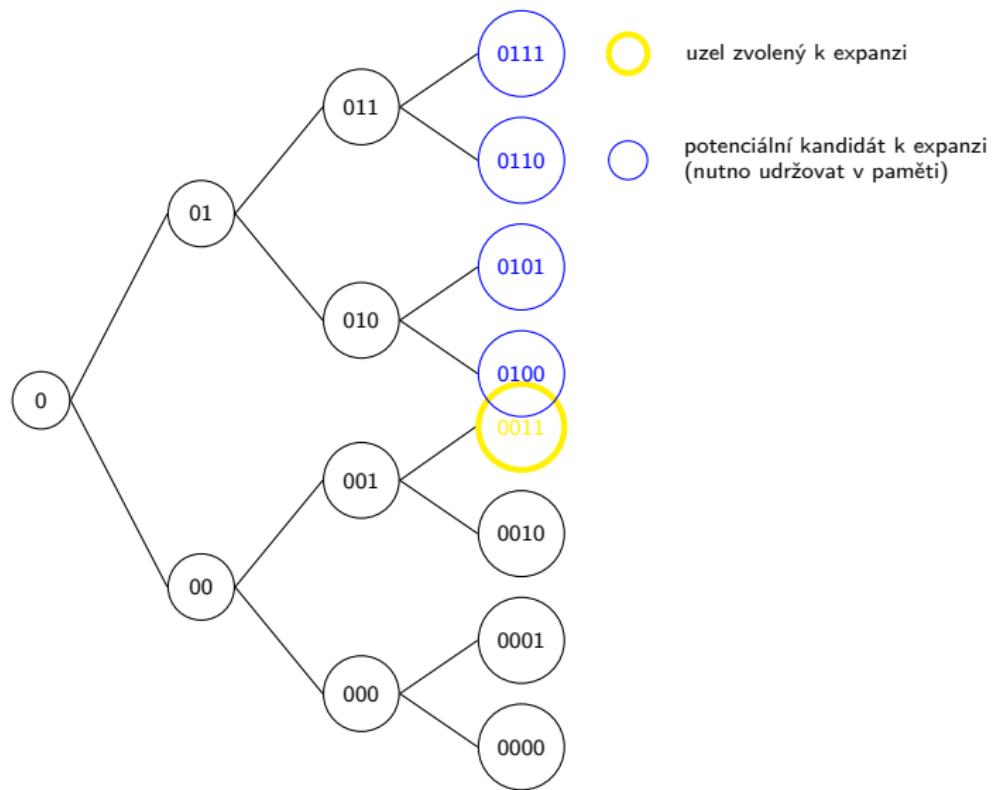
BFS — prohledávání do šířky



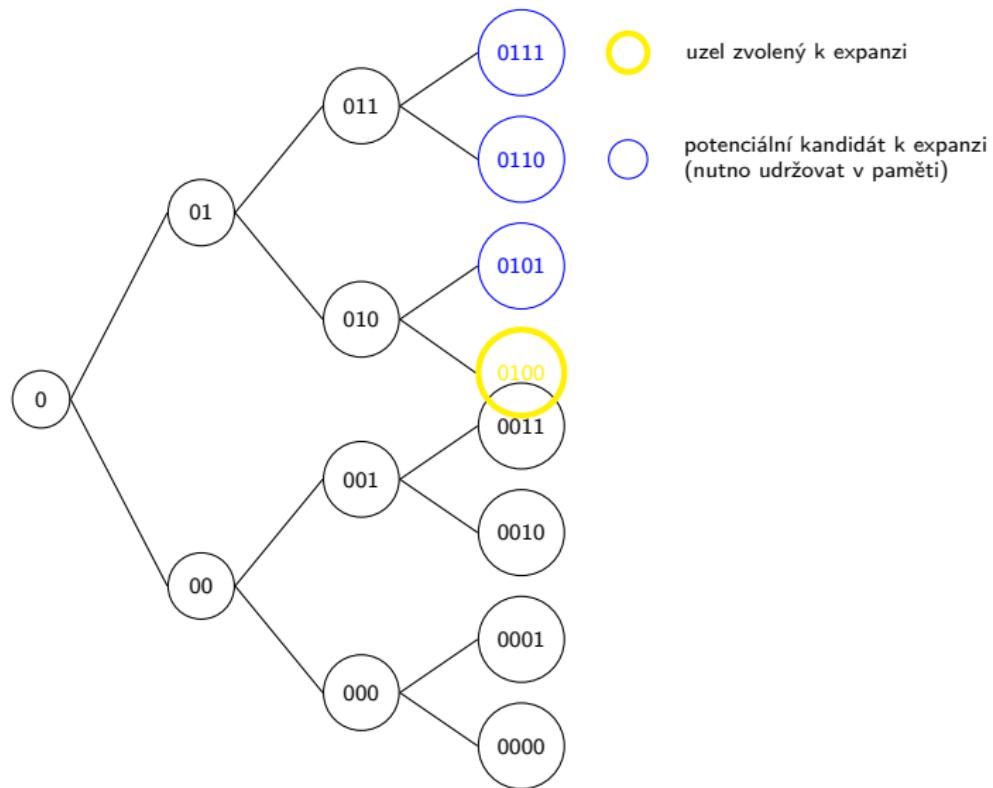
BFS — prohledávání do šířky



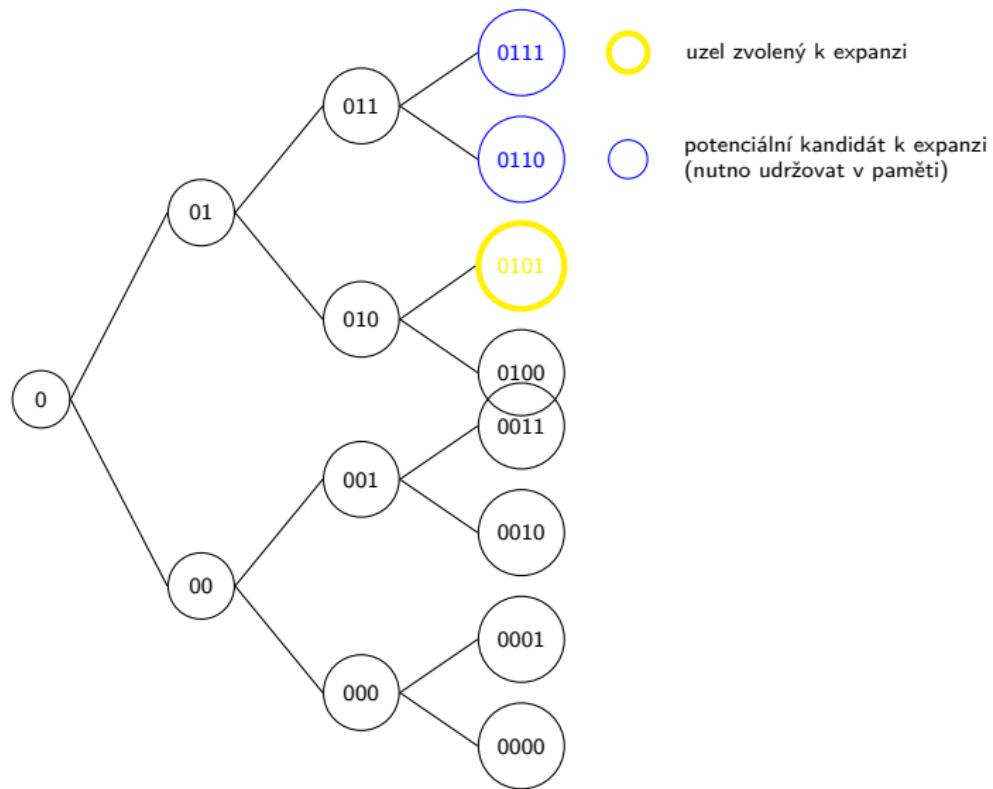
BFS — prohledávání do šířky



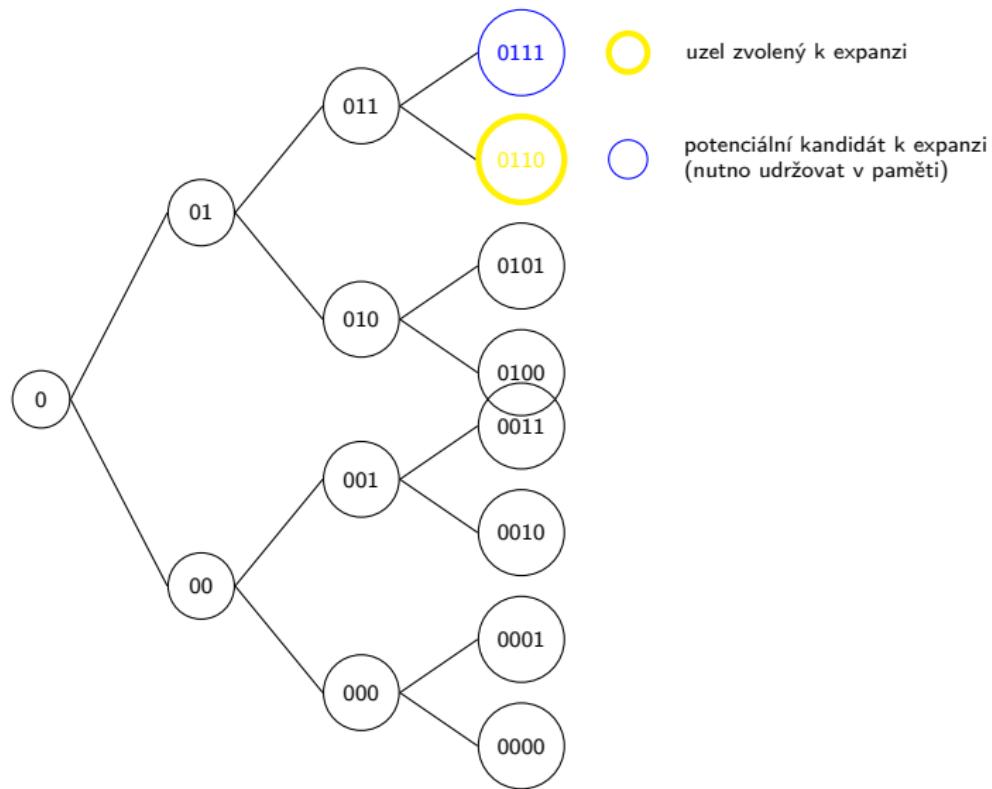
BFS — prohledávání do šířky



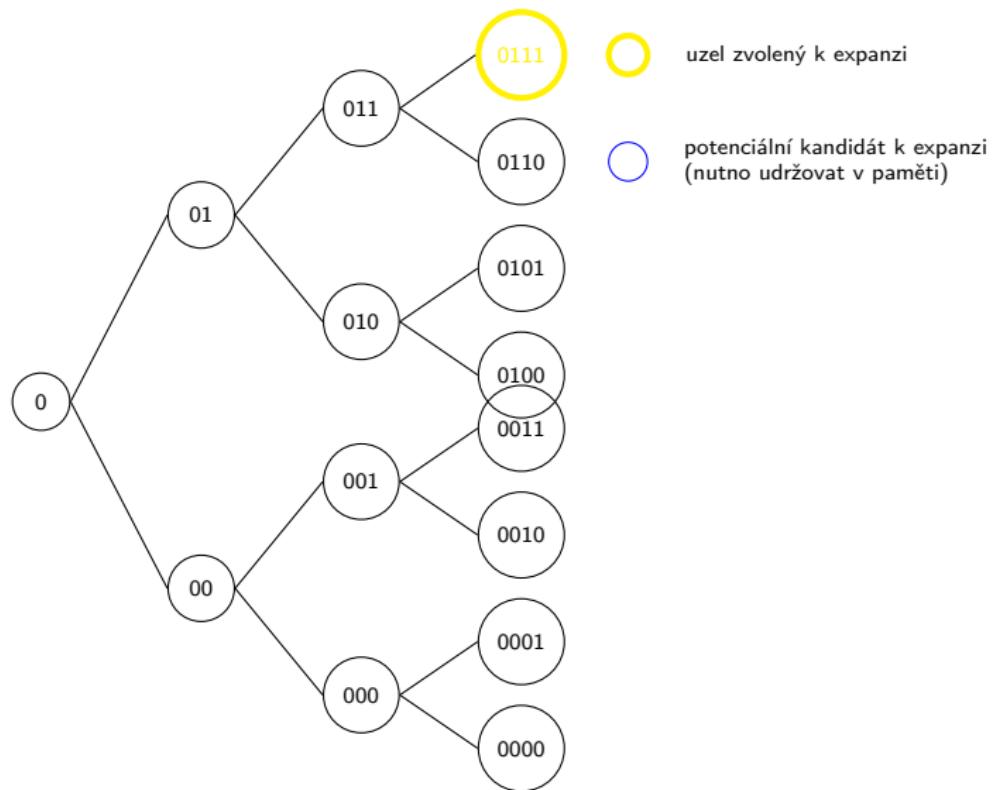
BFS — prohledávání do šířky



BFS — prohledávání do šířky



BFS — prohledávání do šířky



BFS — prohledávání do šířky

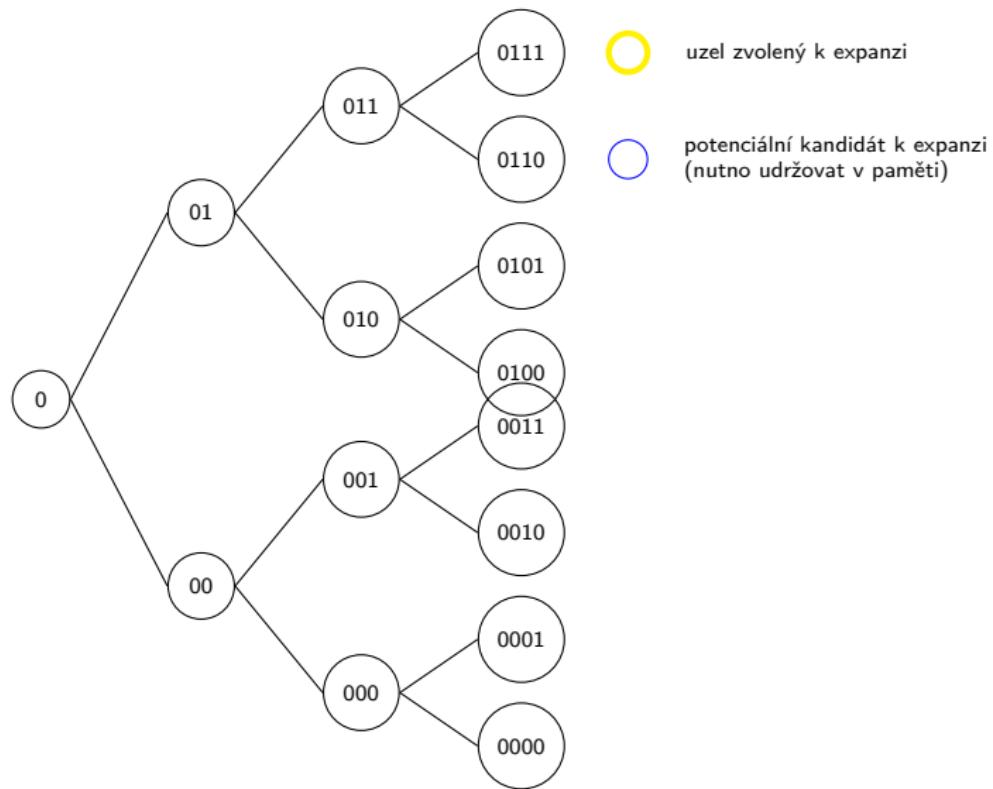
- úplnost: ANO (pokud je b konečné)
- časová složitost: $1 + b + b^2 + \dots + b(b^d - 1) = O(b^{d+1})$
- paměťová náročnost: $O(b^{d+1})$ (celý strom akcí se uchovává v paměti)
- optimalita: to závisí (ano, pokud všechny akce stojí stejně, jinak je třeba modifikovat)

Největším problémem je paměťová náročnost.

$$b = 10, 10000 \text{ uzlů/sec., uzel} = 1\text{Kb}$$

hloubka	uzly	čas	paměť
2	1100	0.1 s	1 Mb
4	111100	11 s	106 Mb
6	10^7	19 min	10 Gb
8	10^9	31 h	1 Tb
10	10^{11}	129 dní	101 Tb
12	10^{13}	35 let	10 Petabyte
14	10^{15}	3523 let	1 Exabyte

DFS — prohledávání do hloubky

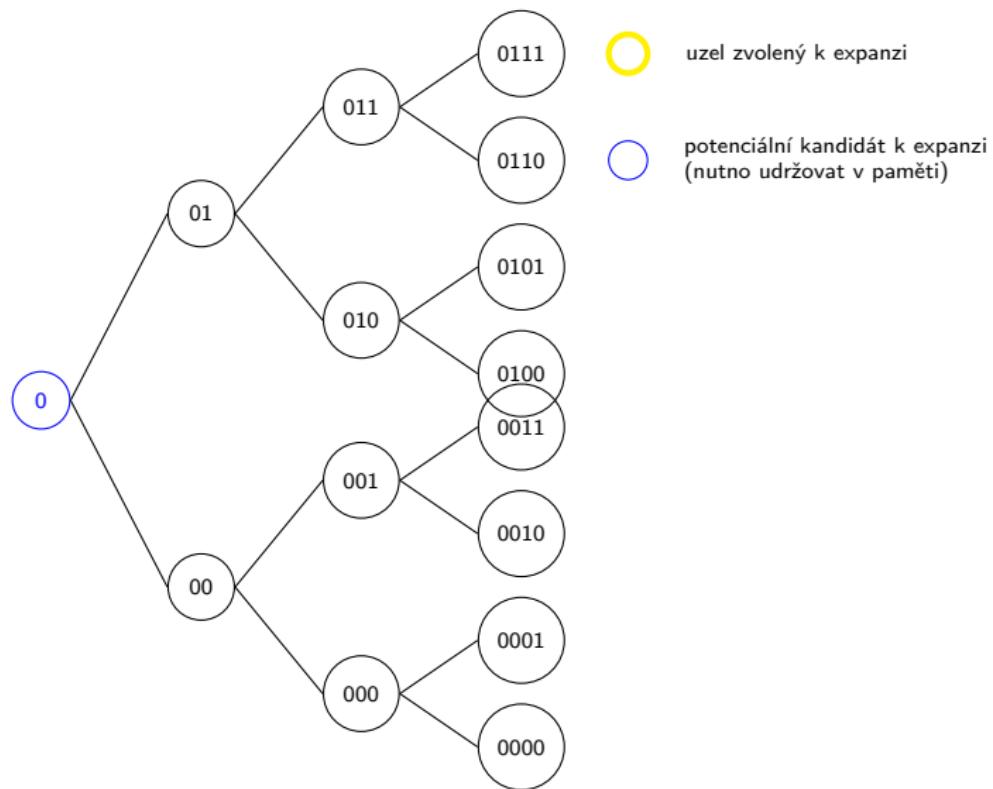


uzel zvolený k expanzi

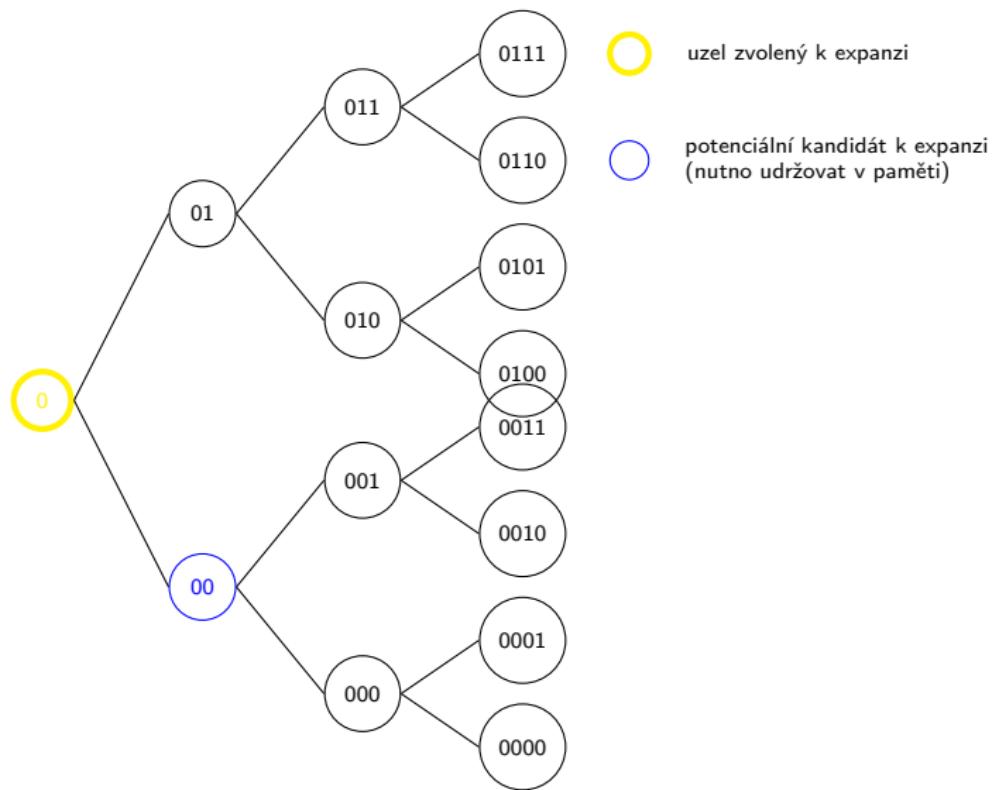


potenciální kandidát k expanzi
(nutno udržovat v paměti)

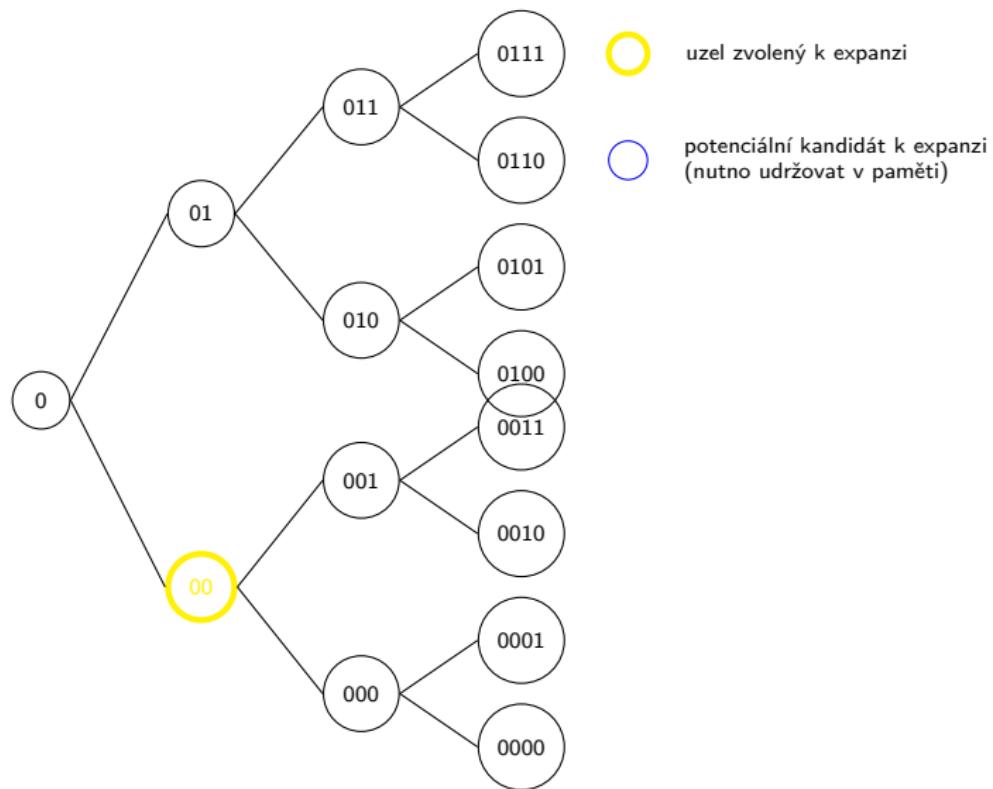
DFS — prohledávání do hloubky



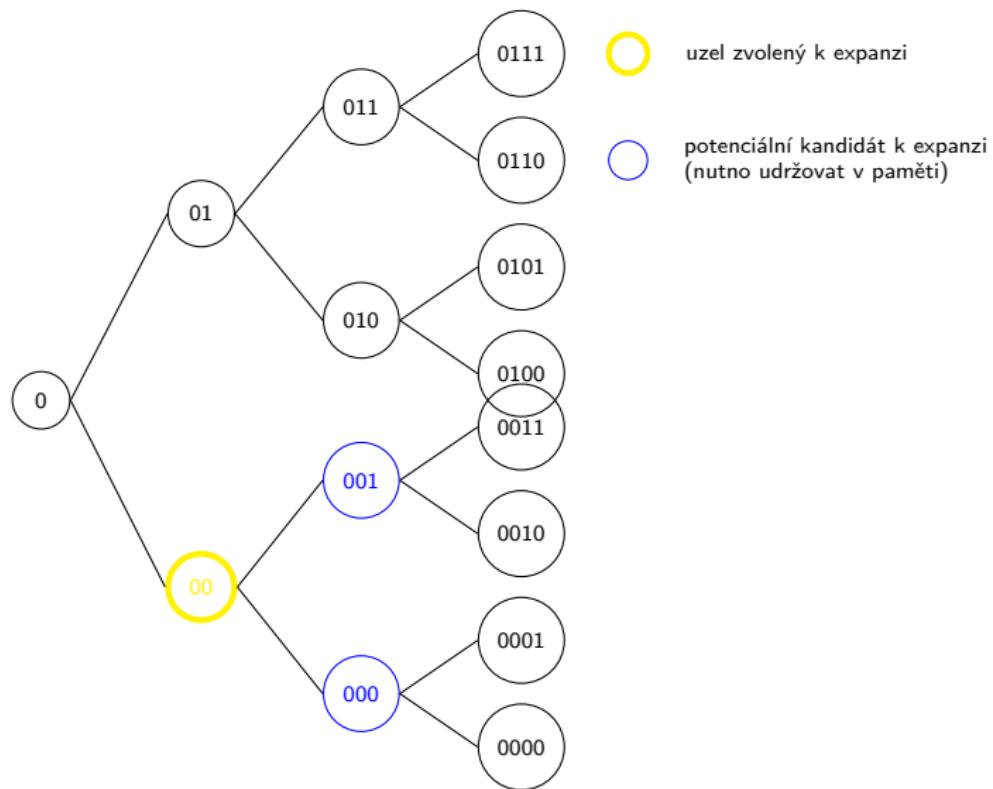
DFS — prohledávání do hloubky



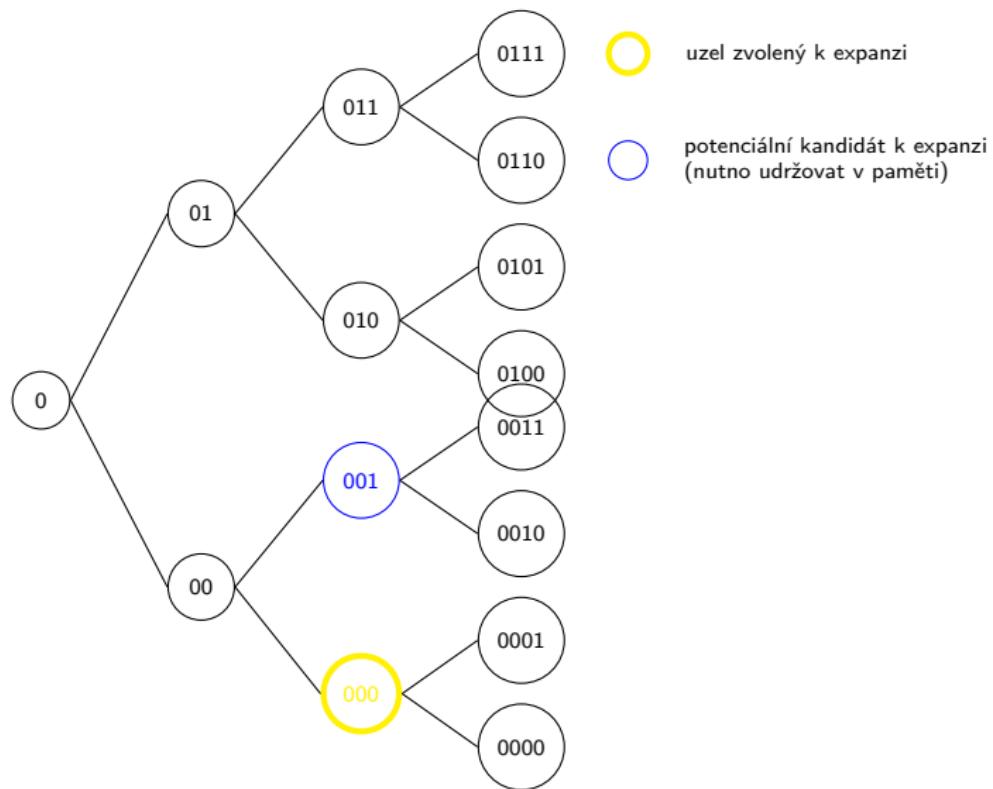
DFS — prohledávání do hloubky



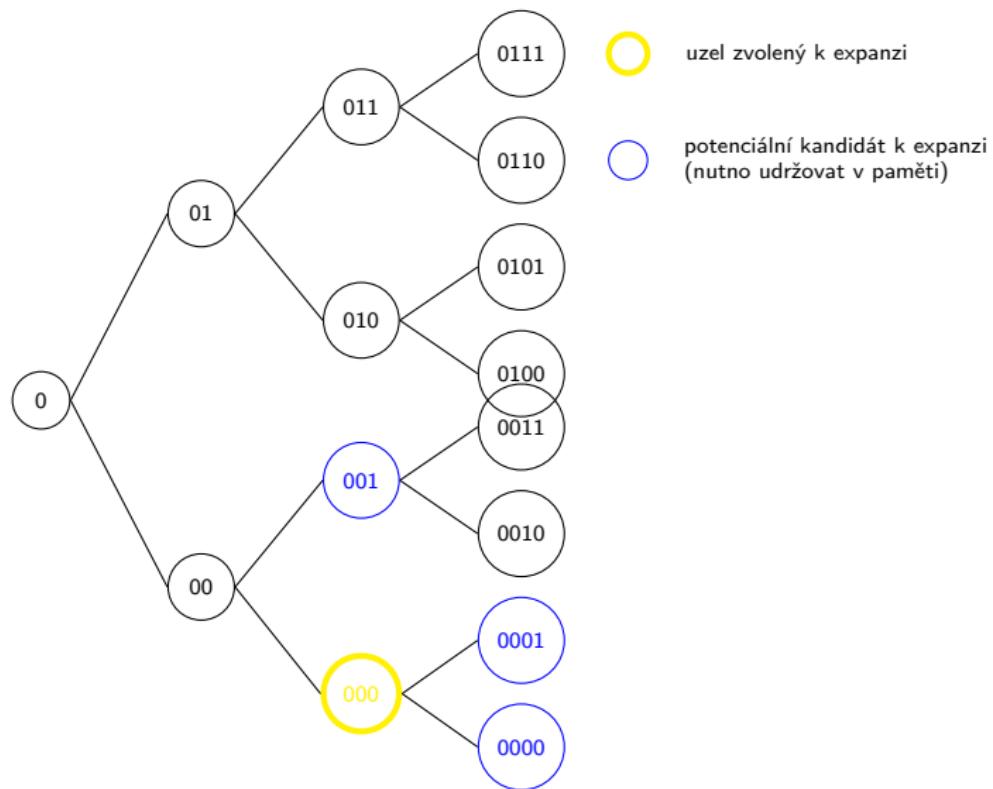
DFS — prohledávání do hloubky



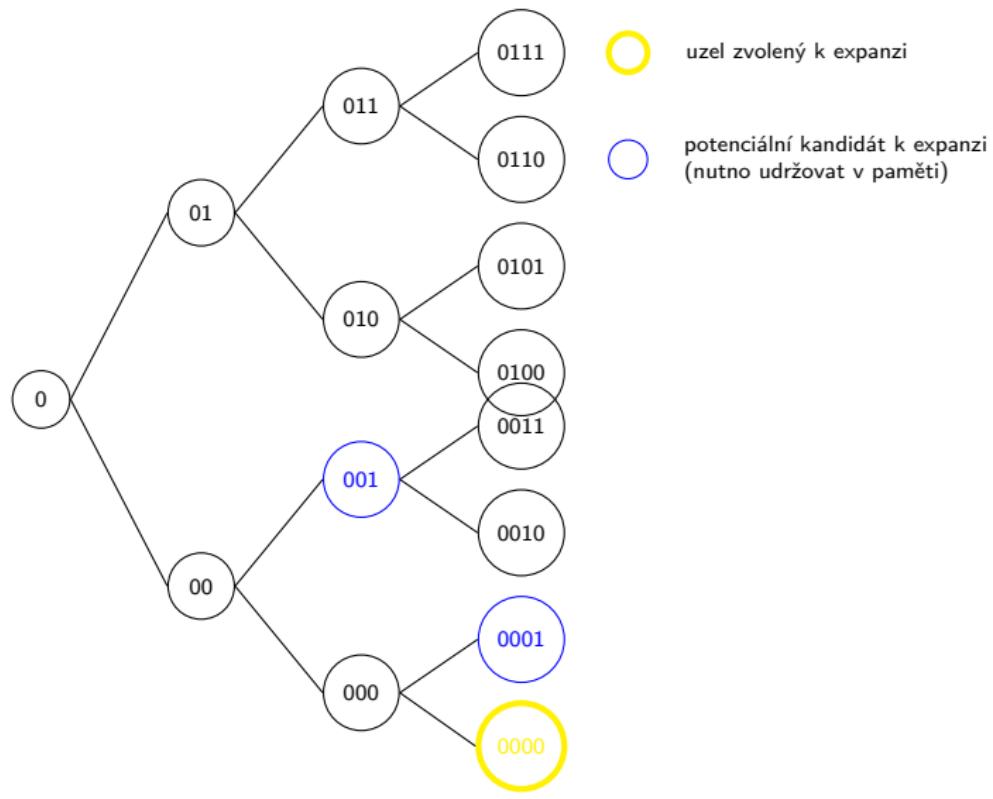
DFS — prohledávání do hloubky



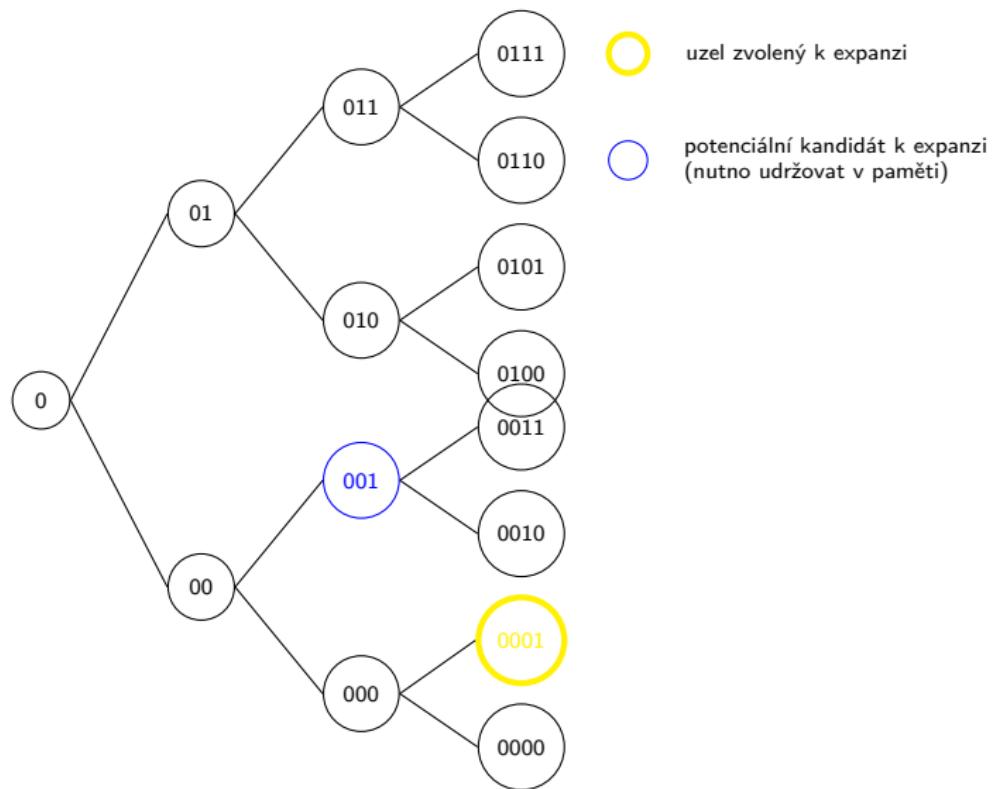
DFS — prohledávání do hloubky



DFS — prohledávání do hloubky



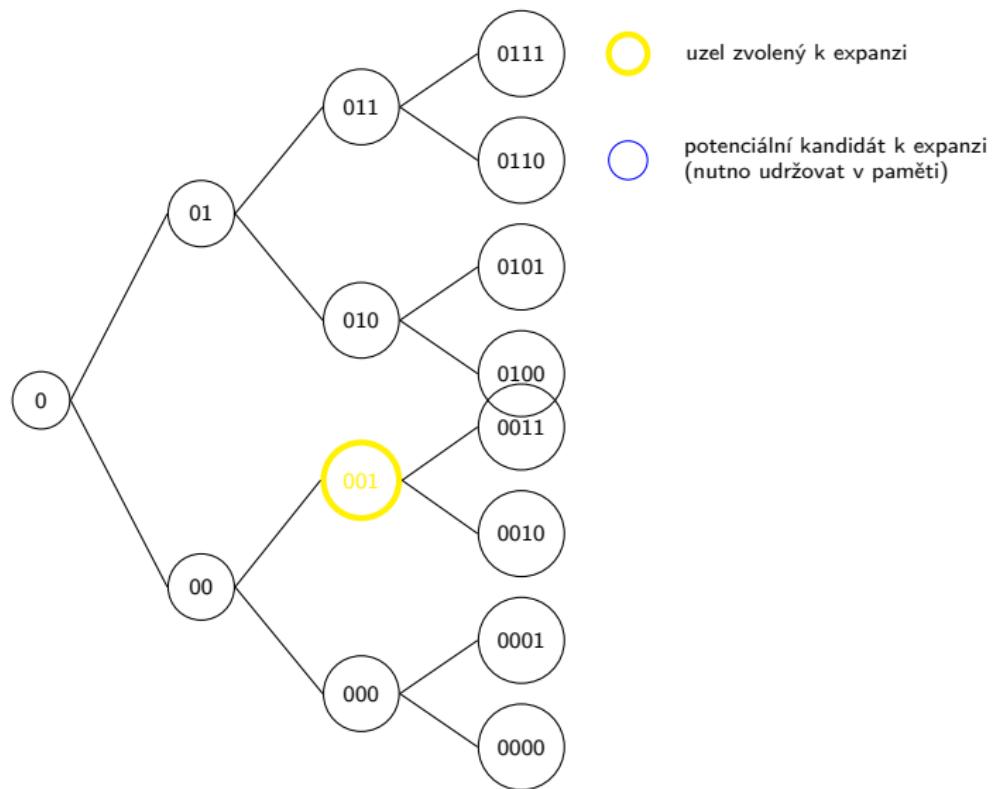
DFS — prohledávání do hloubky



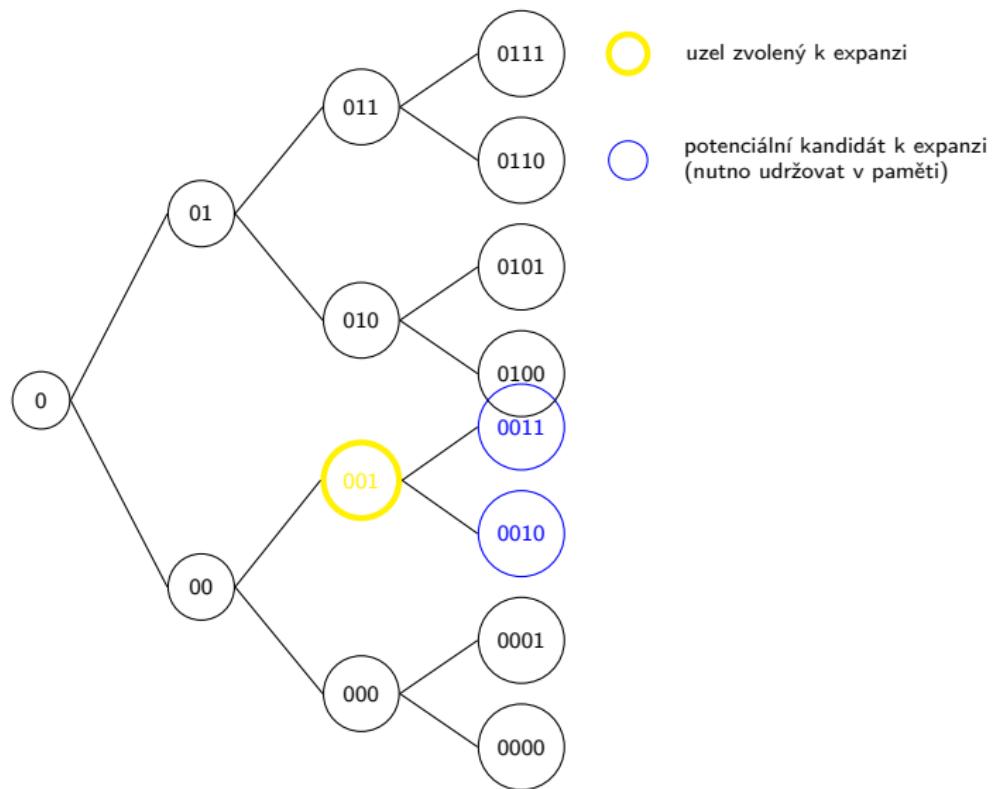
uzel zvolený k expanzi

potenciální kandidát k expanzi
(nutno udržovat v paměti)

DFS — prohledávání do hloubky



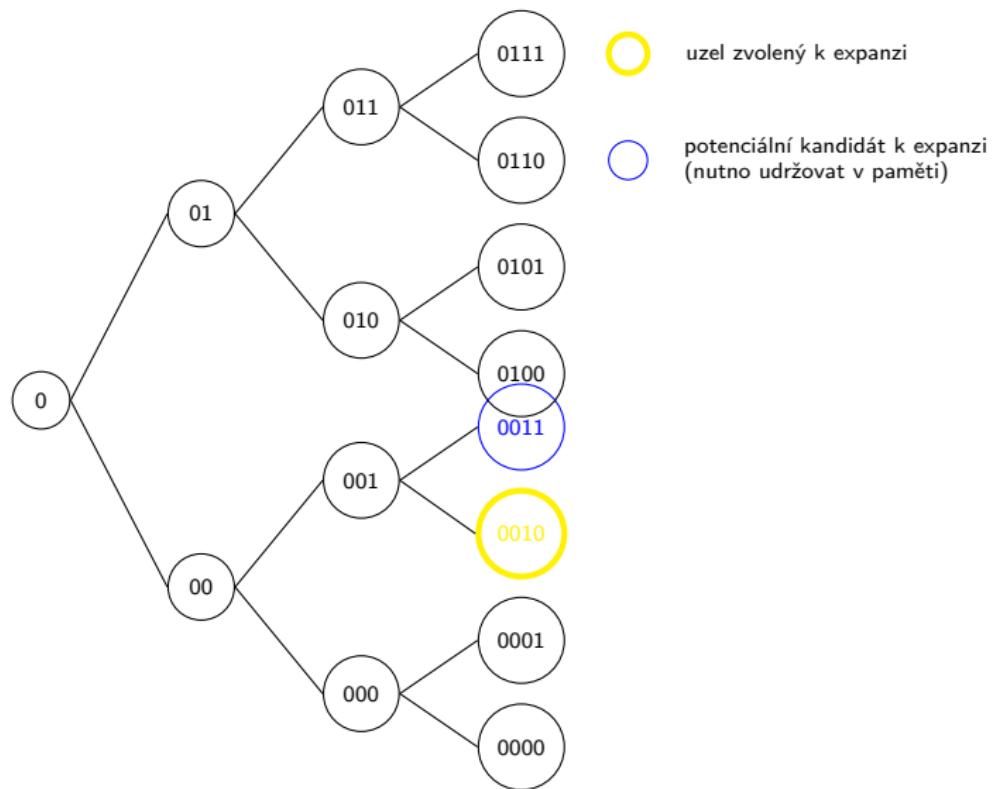
DFS — prohledávání do hloubky



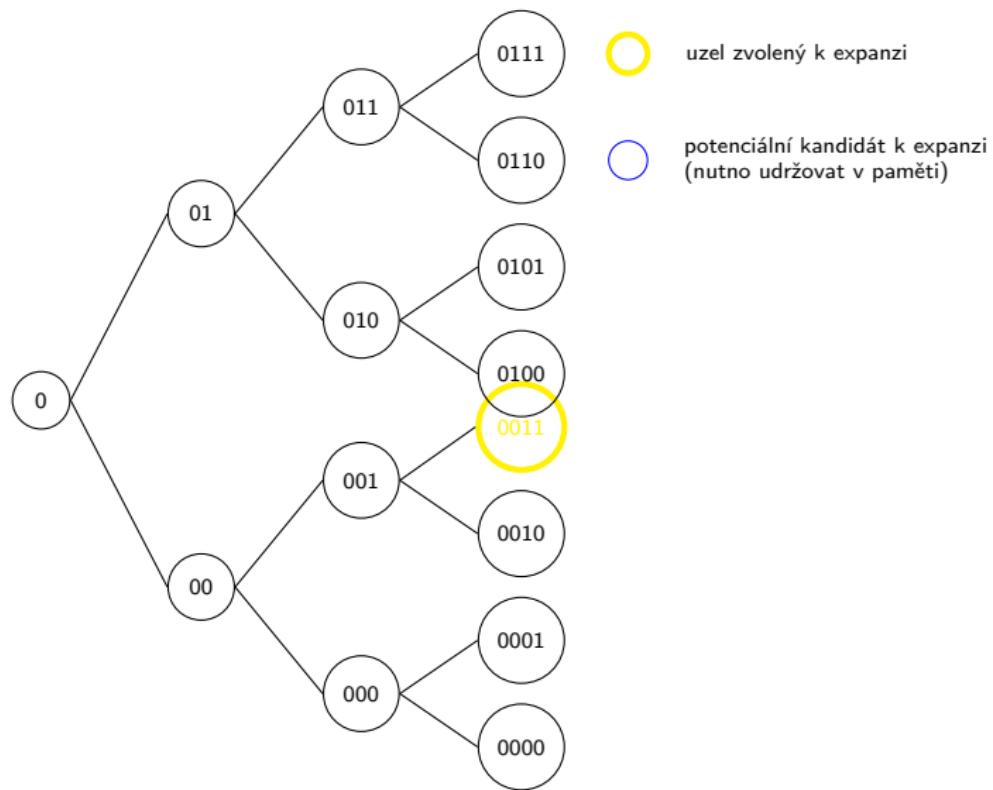
uzel zvolený k expanzi

potenciální kandidát k expanzi
(nutno udržovat v paměti)

DFS — prohledávání do hloubky



DFS — prohledávání do hloubky

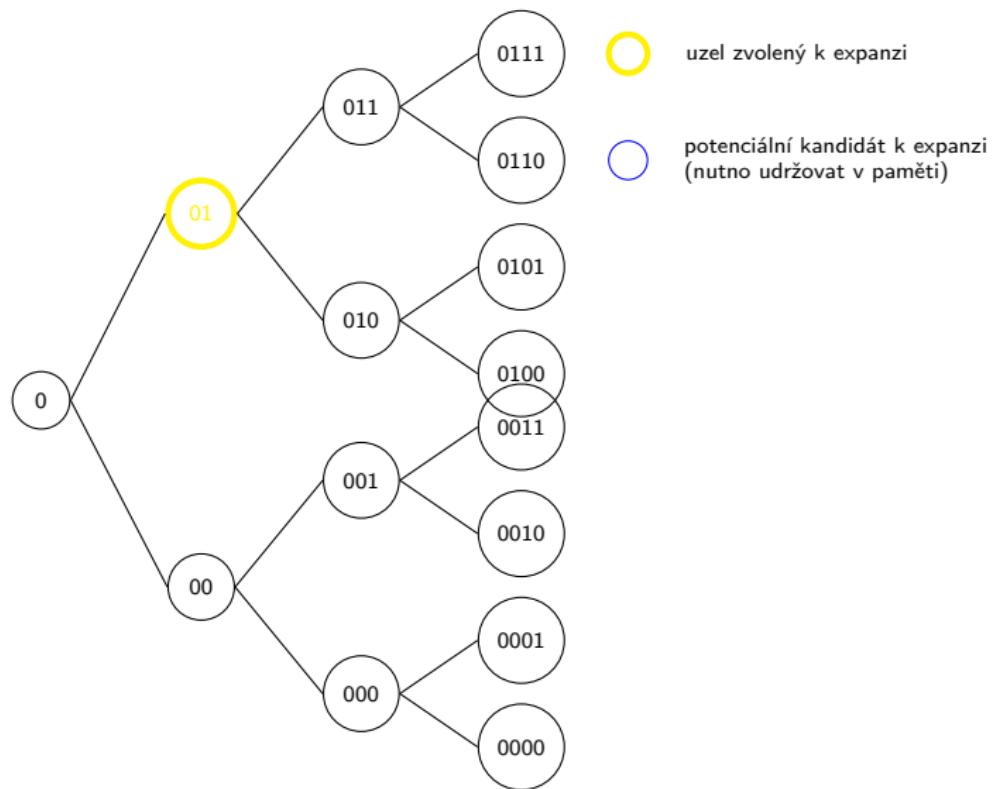


uzel zvolený k expanzi



potenciální kandidát k expanzi
(nutno udržovat v paměti)

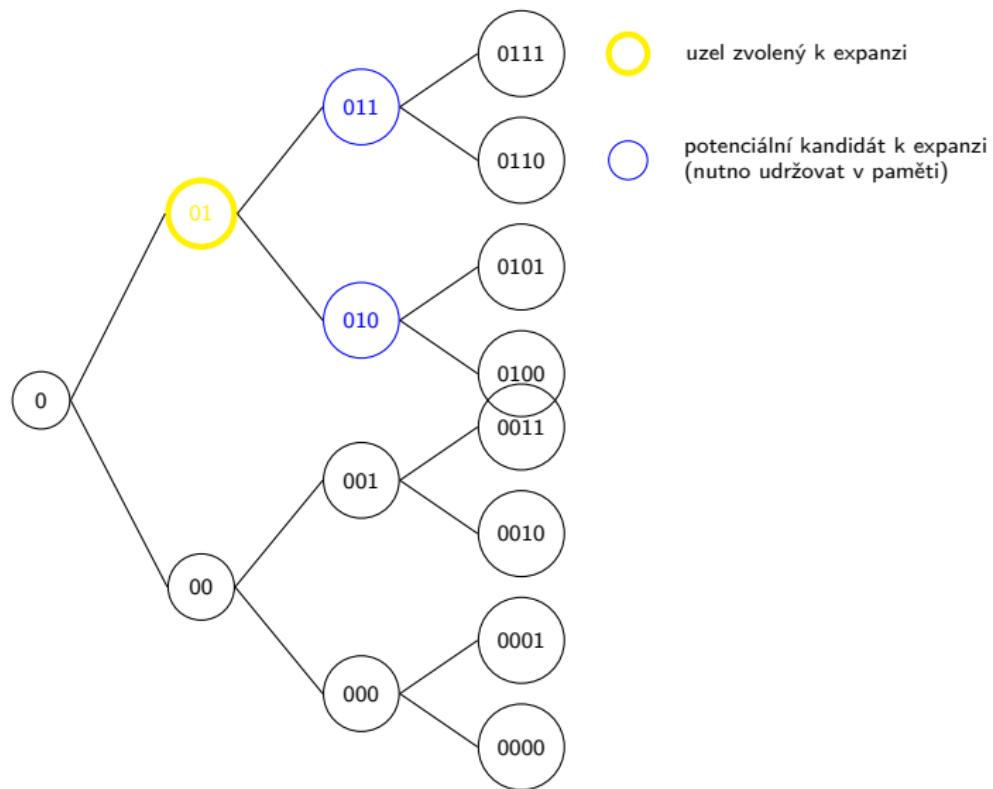
DFS — prohledávání do hloubky



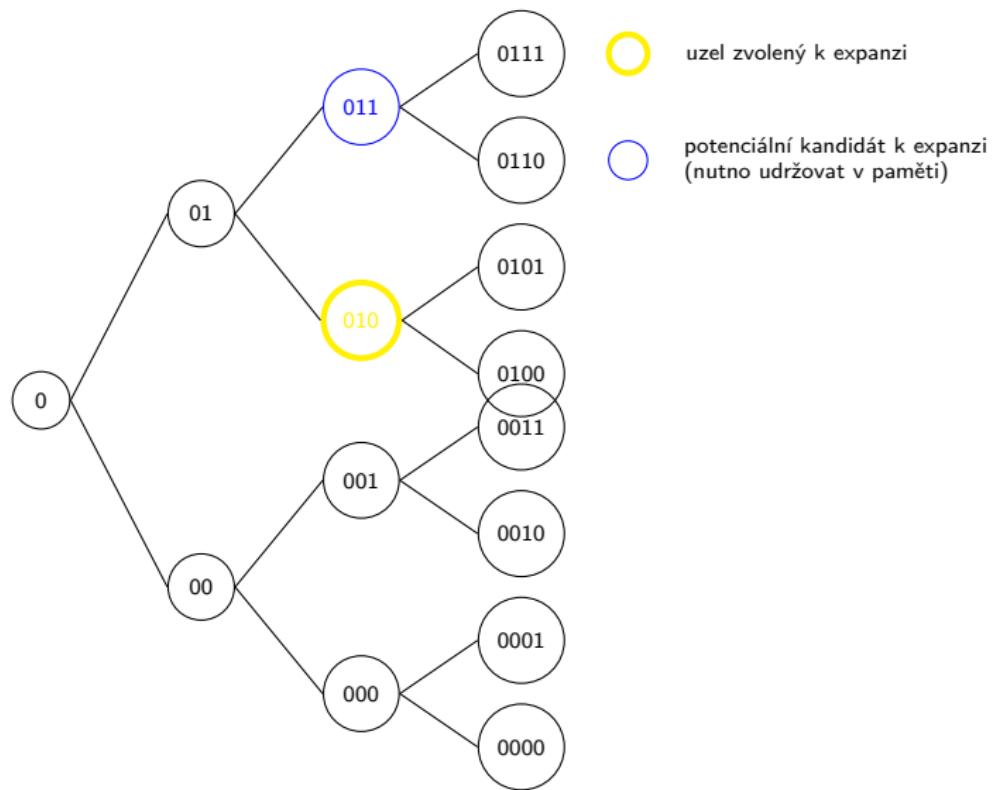
uzel zvolený k expanzi

potenciální kandidát k expanzi
(nutno udržovat v paměti)

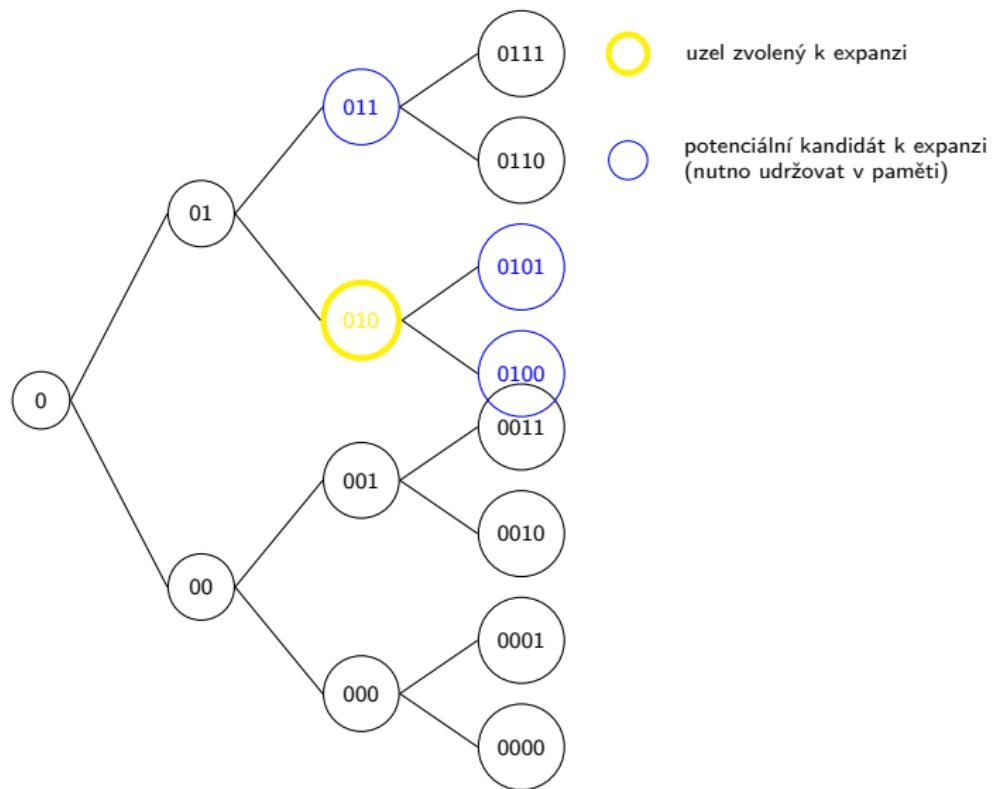
DFS — prohledávání do hloubky



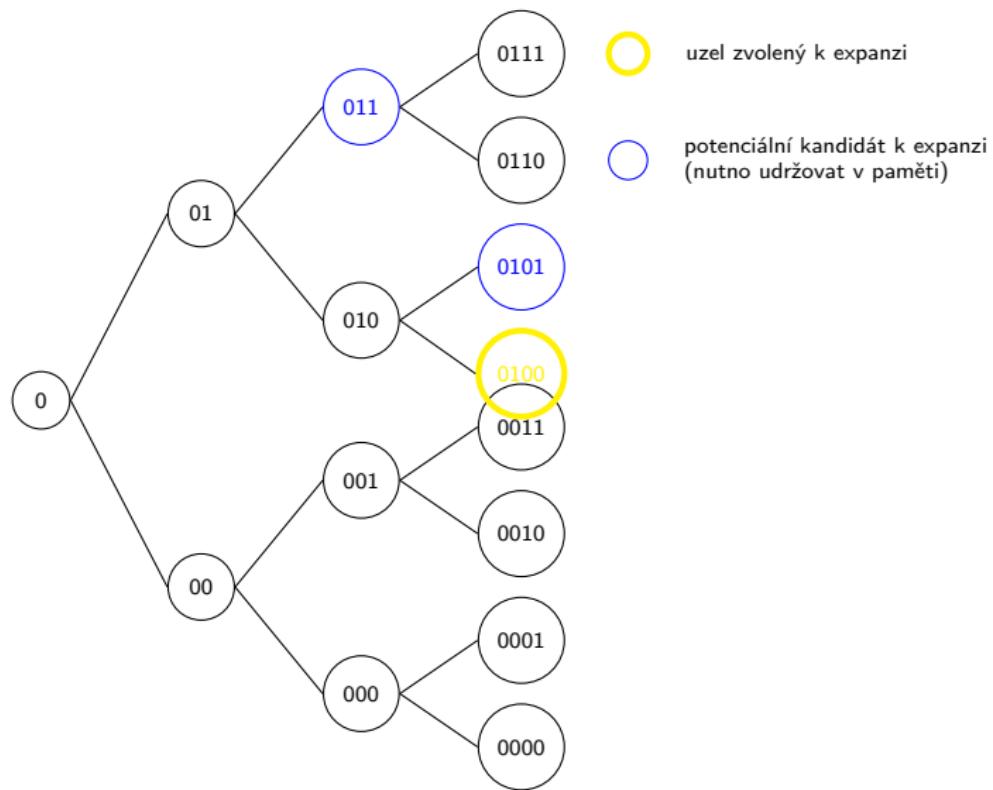
DFS — prohledávání do hloubky



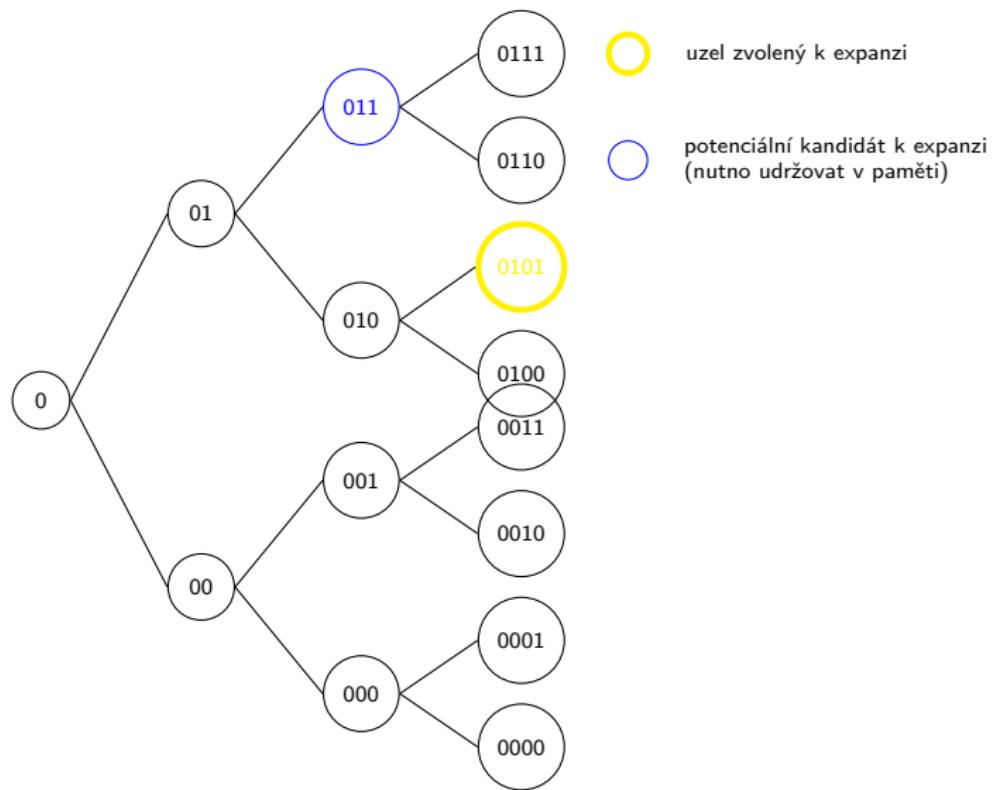
DFS — prohledávání do hloubky



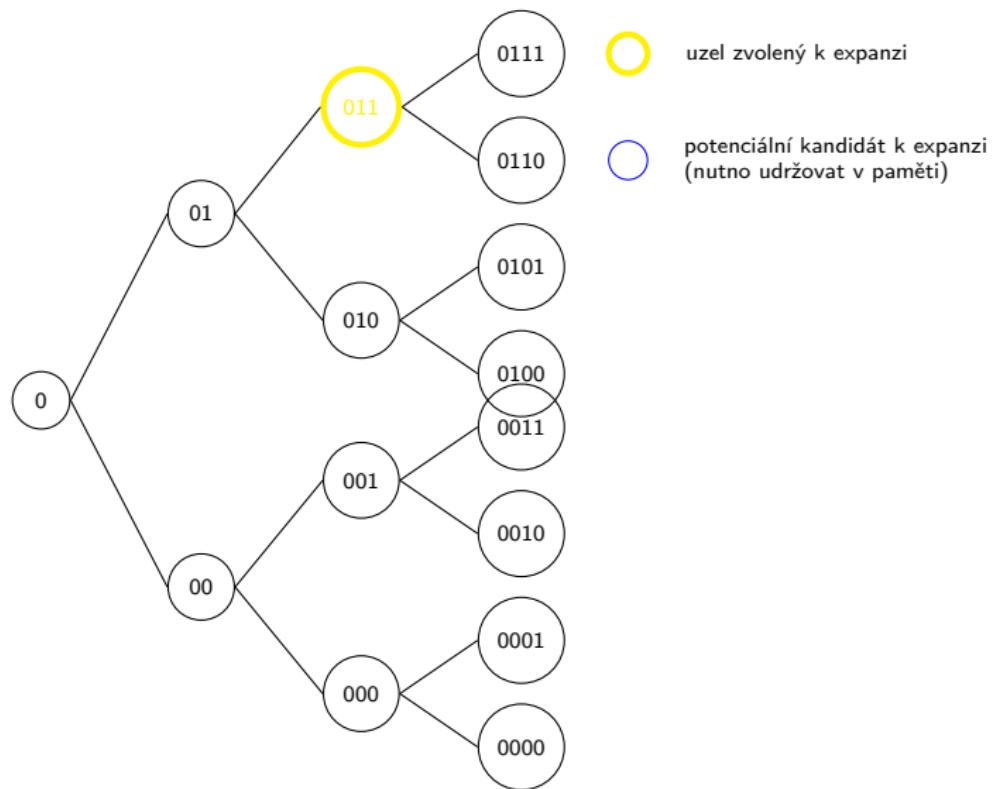
DFS — prohledávání do hloubky



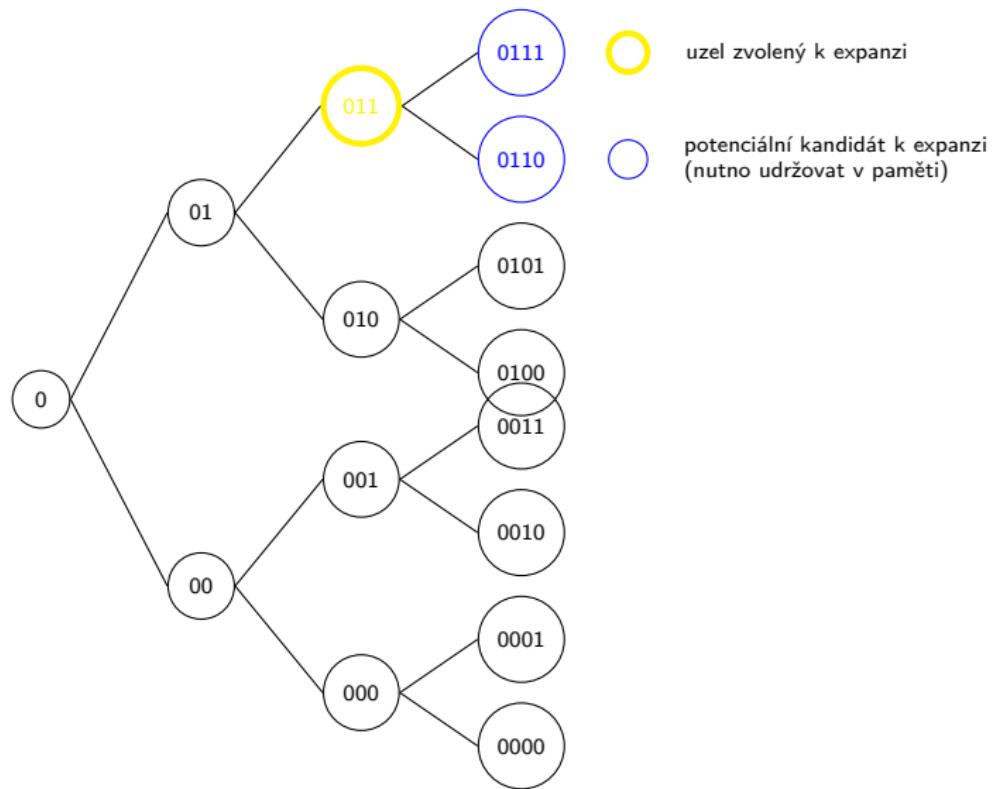
DFS — prohledávání do hloubky



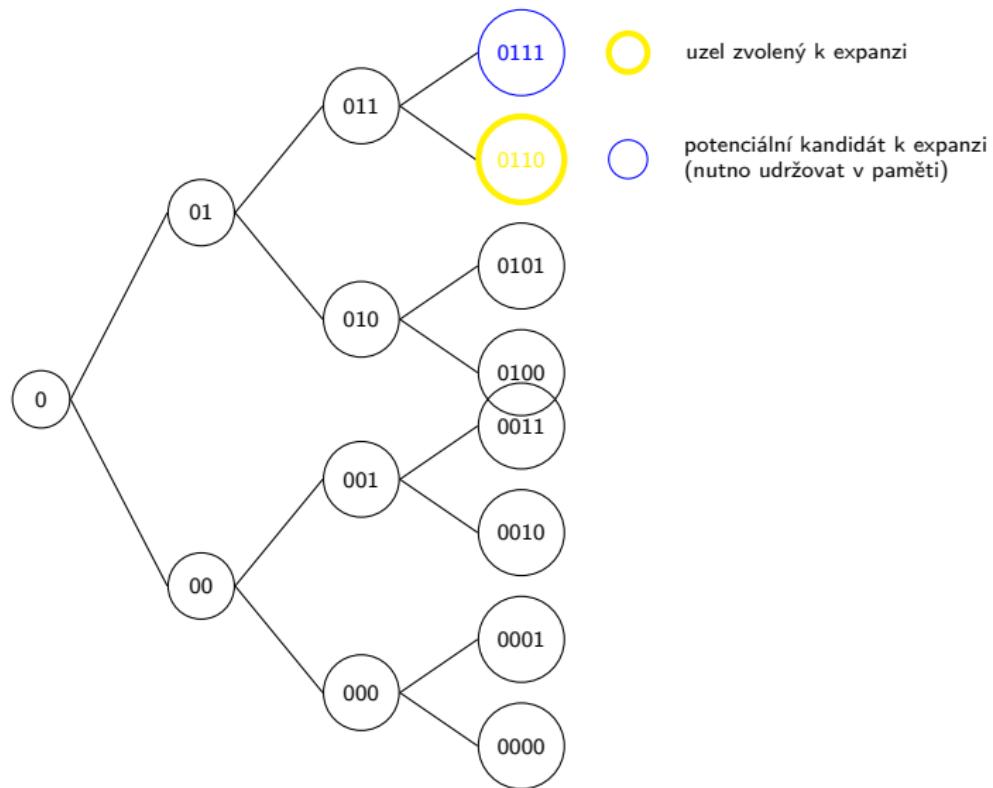
DFS — prohledávání do hloubky



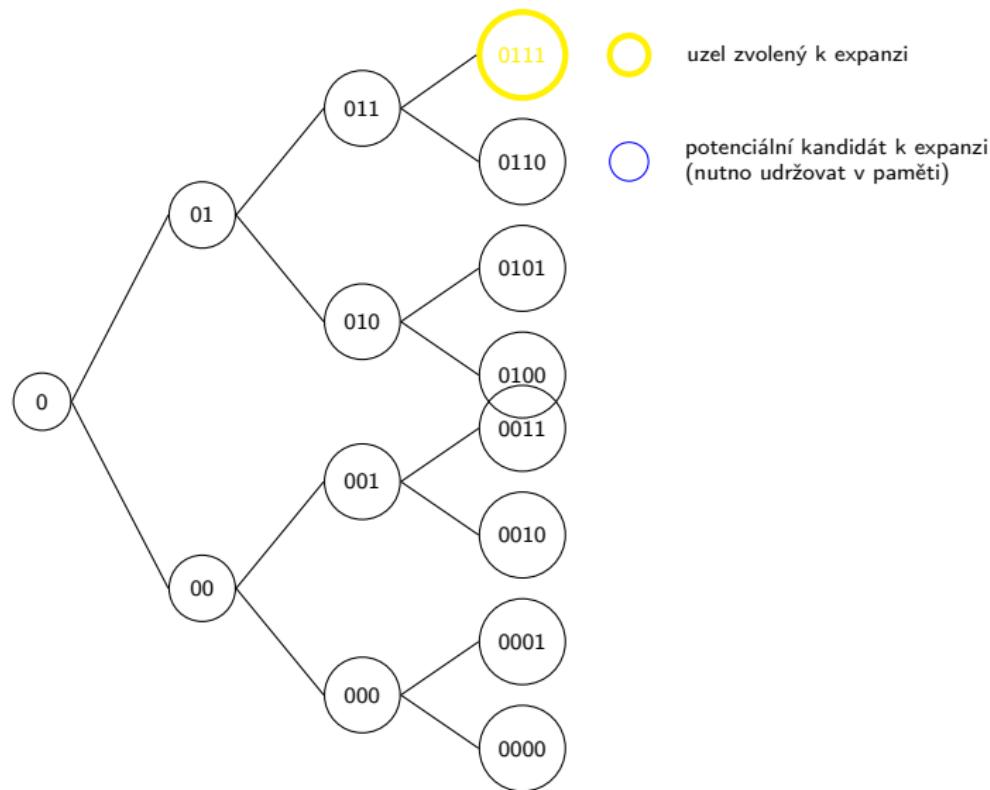
DFS — prohledávání do hloubky



DFS — prohledávání do hloubky



DFS — prohledávání do hloubky



DFS — prohledávání do hloubky

- úplnost: NE (pokud je hloubka nekonečná nebo pokud jsou smyčky)
- časová složitost: $O(b^m)$
 - problém, pokud je $d \ll m$
 - pokud jsou řešení hustě rozložená, je mnohem rychlejší než BFS
- paměťová náročnost: $O(m)$ (**lineární!**)
- optimalita: NE

DLS,IDS — prohledávání do omezené hloubky

DLS — prohledávání do hloubky, ale strom usekneme na hladině l a hlouběji nejdeme.

IDS — postupně aplikujeme DLS s větším a větším l

- úplnost: ANO
- časová složitost: $O(b^d)$
- paměťová náročnost: $O(d)$ (**lineární!**)
- optimalita: ANO (pokud všechny akce stojí stejně, jinak je třeba modifikovat)

Prohledávání z obou konců

Částečně pozorovatelné, nedeterminované světy