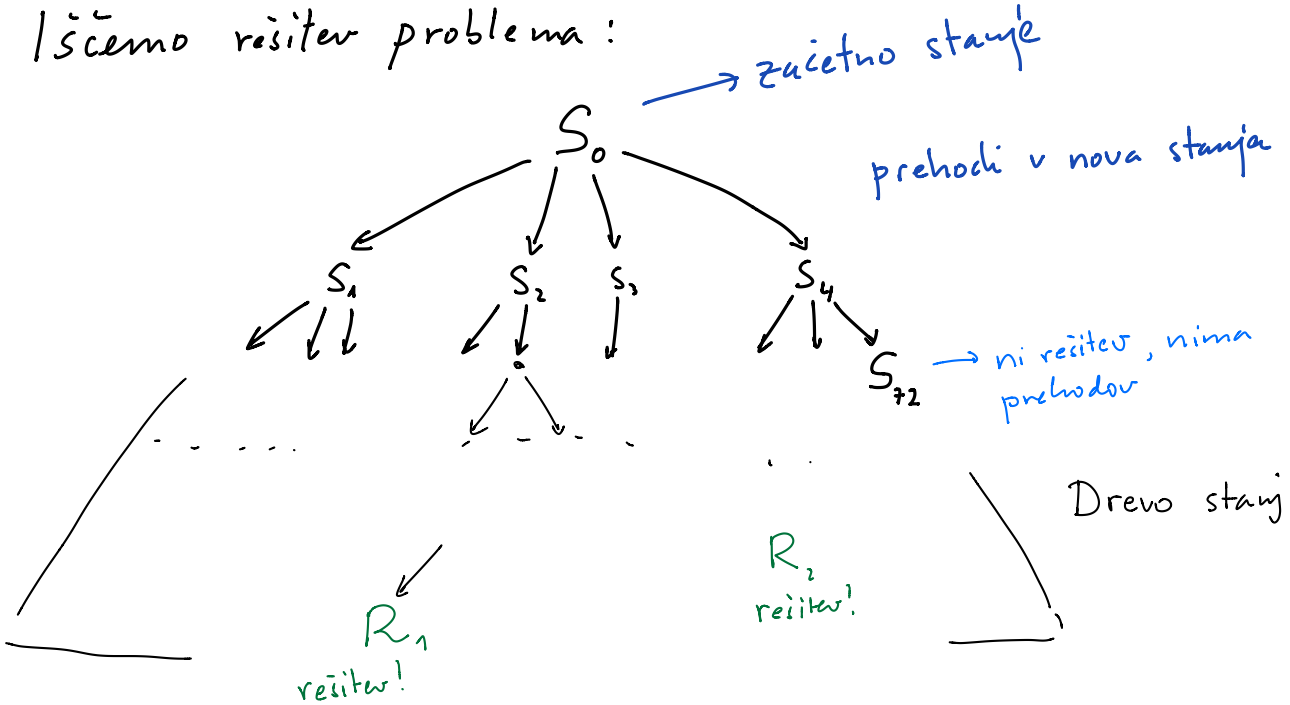
Sestopanje

(angl. backtracking)

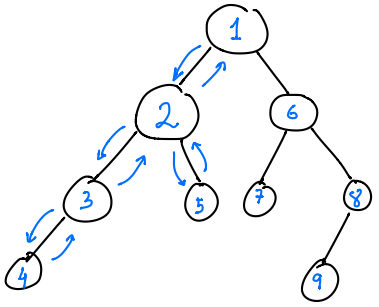
Iščemo rešitev problema:



Kako sistematično pregledamo tako drevo?

Pregled v globino: najprej preiščemo potomce, oz. vedno celotno poddrevo

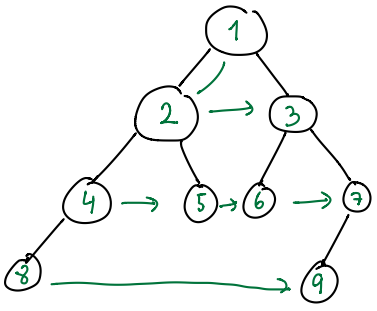
depth-first search (DFS)



Pregled v širino:

breadth-first search (BFS)

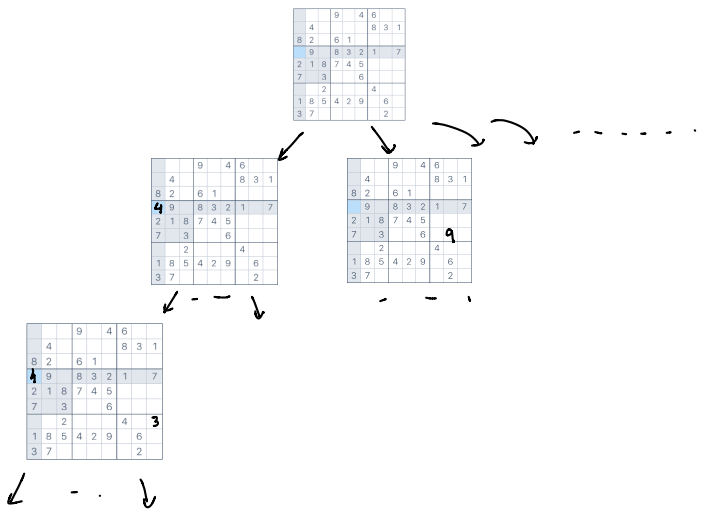
pregledujemo po nivojih



Primeri:

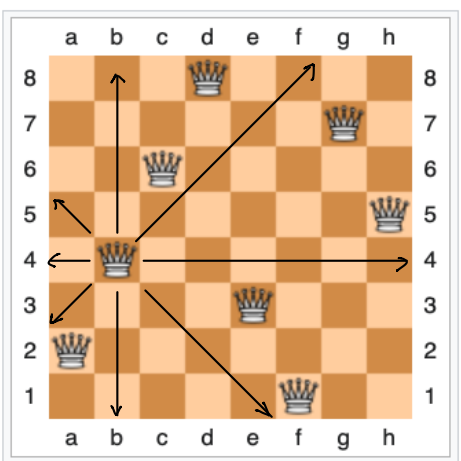
① Sudoku

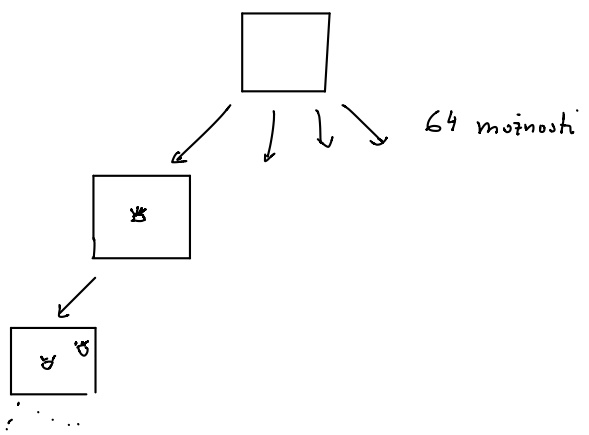
			9		4	6		
	4					8	3	1
8	2		6	1				
	9		8	3	2	1		7
2	1	8	7	4	5			
7		3			6			
		2				4		
1	8	5	4	2	9		6	
3	7						2	



② Šahovske dame

Na šahovnico razporedimo 8 dam, da se med seboj ne napadajo.





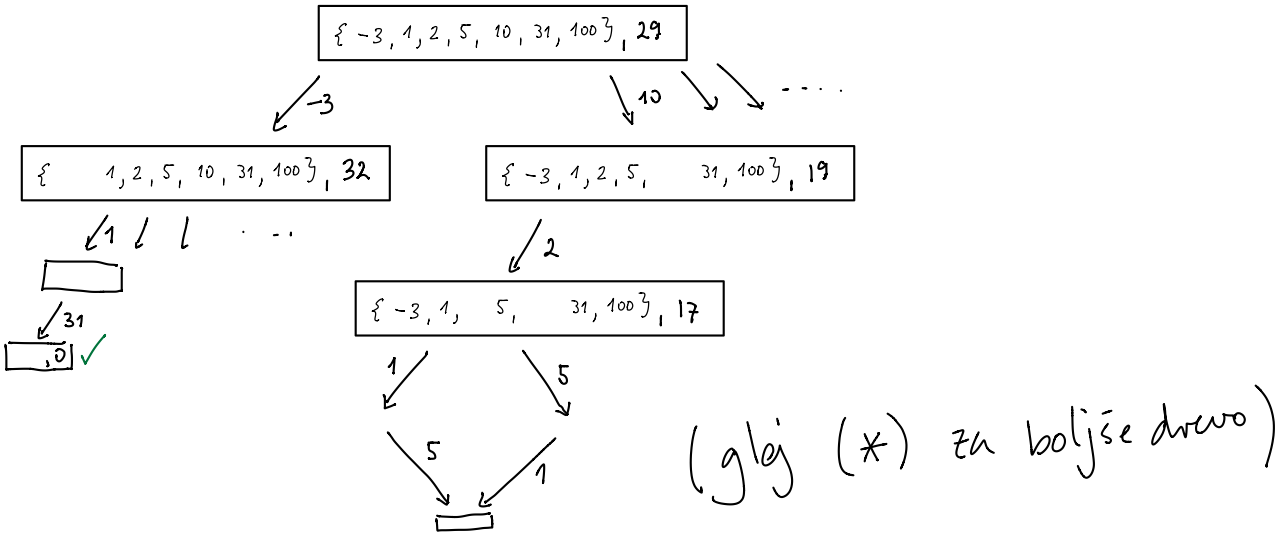
③ Problem vsote podmnožice (subset-sum problem):

- Imamo množic celih števil $S \subseteq \mathbb{Z}$ in ciljno vsoto $k \in \mathbb{Z}$
- Naloga: ali obstaja $T \subseteq S$, da je vsota elementov T enaka k?

$S = \{-3, 1, 2, 5, 10, 31, 100\}$ $\quad k = 29$

odgovor: $T = \{31, -3, 1\}$

Iskanje rešitve, predstavljeno z drevesom stanj in prehodov:



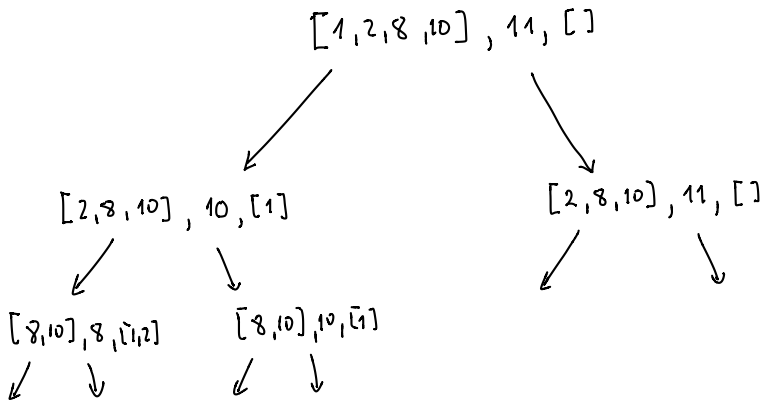
(glej (*) za boljše drevo)

Kako to naprogramiramo?

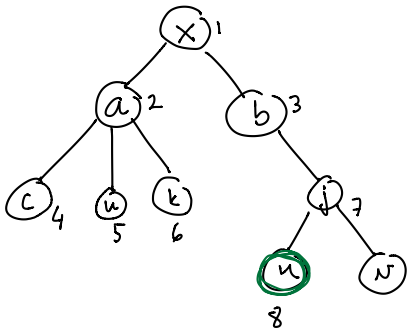
① Drevesa ne hranimo kot podatek, sproti ga gradimo:

- → imamo trenutno stanje S
- → izračunamo možne prehode iz S: $S_1 \dots S_n$ in jih pregledamo

(*)



Kako naprogramiramo pregled v širino ?



vrsta:
- x
- a, b
  b, c, u, k
  c, u, k, j
  u, k, j
  k, j
  .
  j
  u, v

VRSTA stanj, kei jih moramo še obravnavati

Na začetku v vrsti stoji x:

x

Splošni korak:

$s_i, s_j, s_k, \ldots, s_n$

čakajo na obravnavo

ta je na vrsti:

rešitev s_i?

da ✓

ne

naslednike $t_1, \ldots t_m$ stanja s_i postavimo na konec vrste

$s_j, s_k, \ldots, s_n, t_1, \ldots, t_m$

Sredi pregleda v širino: