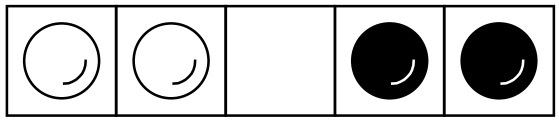
Well-defined formulation of the stone puzzle:



<u>states</u>: position of each stone? *not* good →

square content - 5 variables, 3 values each
white (O), black (X), empty (–)

<u>initial state</u>: (O O – X X)

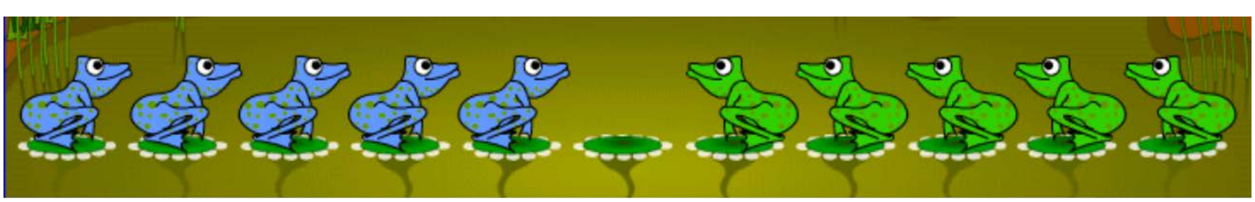
<u>goal test</u>: state equal to (X X – O O)

<u>operators</u>: details? (O O – X X) → (O – O X X) ?!?
too many → *need to abstract*

- MoveToRight: (O –) → (– O)
- MoveToLeft: (– X) → (X –)

- JumpToRight: (O X –) → (– X O)
- JumpToLeft: (– O X) → (X O –)

<u>path cost</u>: number of operators used (1 for all ops)



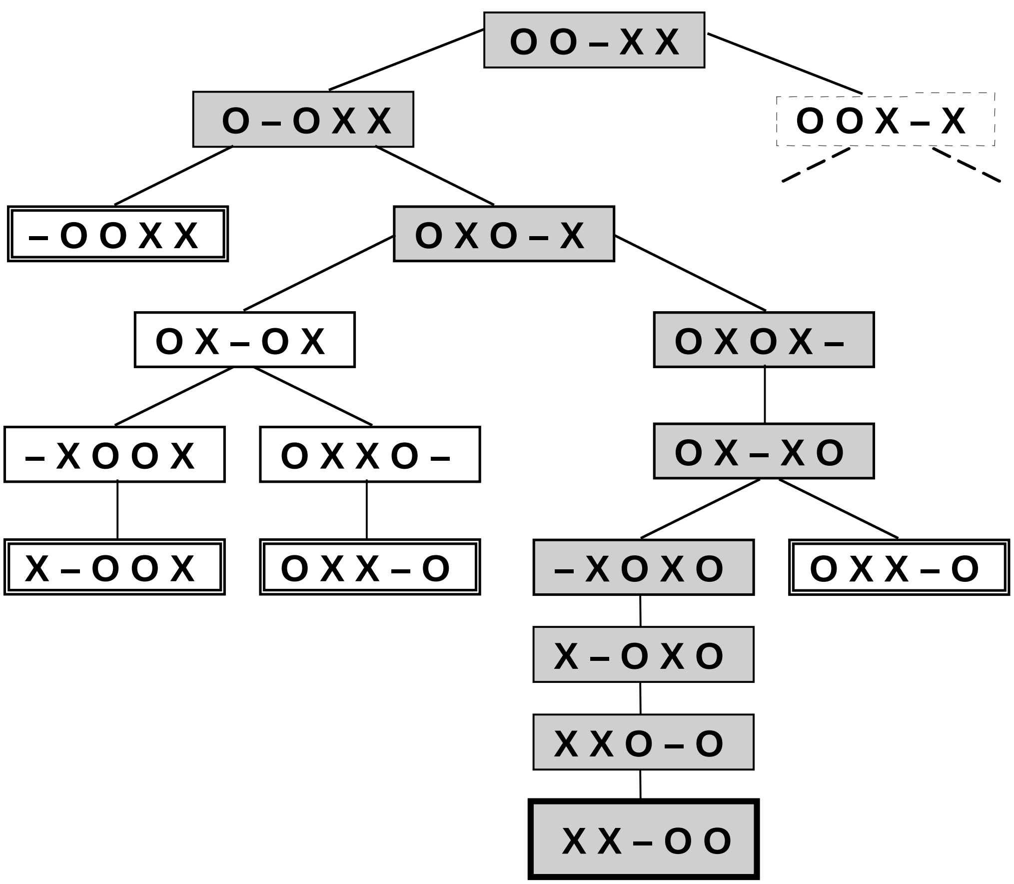
Problem search tree and solution:

- valid, reachable states only (subset of the state space)
- symmetric portion of the search tree not shown

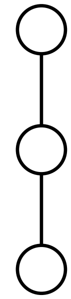
init: (O O – X X)

ops: - MR: (O –) → (– O) - JR: (O X –) → (– X O)

- ML: (– X) → (X –) - JL: (– O X) → (X O –)



Characteristics of the search space:



nb of branches: $2 * 15 = 30$

non-terminal nodes: $1 + 2 * 10 = 21$

<u>average branching factor</u>: $30 / 21 \approx \underline{1.43}$

<u>depth</u> of the 2 solutions: <u>8</u>

<u>space complexity</u>:

- actual space required = <u>31</u> nodes
- theoretical = $1 + 1.43 + 1.43^2 + \ldots + 1.43^8 \approx \underline{55}$

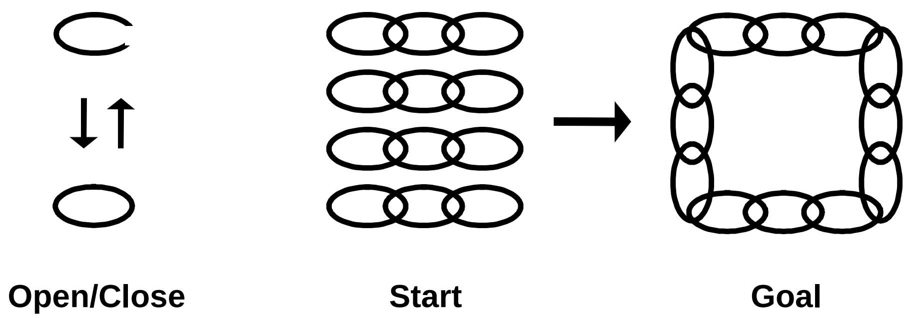
“average” → - not very relevant if search space is small
- based on a *uniform* search tree; *but* here d=8 for solution path and only 5 otherwise!

Most suitable search algorithm:

(note: for small problems, *any* algorithm will do!)

- heuristic function? no → non-informed search (else A*)
- any solution ok? → DFS (else IDS)
- optimal solution? low branching factor → BFS, (else IDS)
  variable operator cost? → UCS

Formulation of the chain problem:



<u>states</u>: - set of n chains
- chains of k links, circular or not (l = 0 or 1)
- links open or closed (c = 0 or 1)

→ { … (k, l, c) … } (note: c=0 for k>1)

<u>initial state</u>: { (3,0,0) (3,0,0) (3,0,0) (3,0,0) }

<u>goal state</u>: { (12,1,0) }

<u>operators</u>: OS: *open a single link*
"*open*"
(1,0,0) → (1,0,1)

OE: *open a link at the end of a chain*
(k,l,0) → (1,0,1) + (k–1,0,0) $k > 1$

OM(m): *open a link in the middle of a chain*
(k,0,0) → (1,0,1) + (m,0,0) k > 2
+ (k–m–1,0,0) k–1>m>0

operators: CS: *close a single link*
“*close*” (1,0,1) → (1,0,0)

CE(*l*): *close a link at the end of a chain*
(1,0,1) + (*k*,0,0) → (*k*+1,*l*,0)

CM: *close a link in between two chains*
(*k*,0,0) + (*m*,0,0) + (1,0,1) → (*k*+*m*+1,0,0)

- note: can be abstracted to O() and C() only

path cost: number of operators applied (1 for all ops)

Optimal solution to the chain problem:

$\{ (3,0,0), (3,0,0), (3,0,0), \underline{(3,0,0)} \}$

OM(1): $\{ (3,0,0), (3,0,0), (3,0,0), (1,0,1), \underline{(1,0,0)}, (1,0,0) \}$

OS(): $\{ (3,0,0), (3,0,0), (3,0,0), (1,0,1), (1,0,1), \underline{(1,0,0)} \}$

OS(): $\{ \underline{(3,0,0)}, \underline{(3,0,0)}, (3,0,0), \underline{(1,0,1)}, (1,0,1), (1,0,1) \}$

CM(): $\{ \underline{(7,0,0)}, \underline{(3,0,0)}, \underline{(1,0,1)}, (1,0,1) \}$

CM(): $\{ \underline{(11,0,0)}, \underline{(1,0,1)} \}$

CE(1): $\{ (12,1,0) \}$ *6 steps only*