Discussion

CS 5/7320
Artificial Intelligence

Solving problems by searching

AIMA Chapter 3

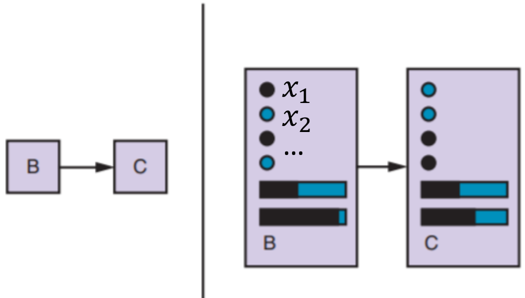
Slides by Michael Hahsler
based on slides by Svetlana Lazepnik
with figures from the AIMA textbook.

State Space for Search

State Space

- Number of different states the agent and environment can be in.
- **Reachable states** are defined by the initial state and the transition model. Not all states may be reachable from the initial state.
- **Search tree** spans the state space. Note that a single state can be represented by several search tree nodes if we have redundant paths.
- State space size is an indication of problem size.

State representation



(a) Atomic (b) Factored

The state consists of variables called fluents that represent conditions that can change over time.

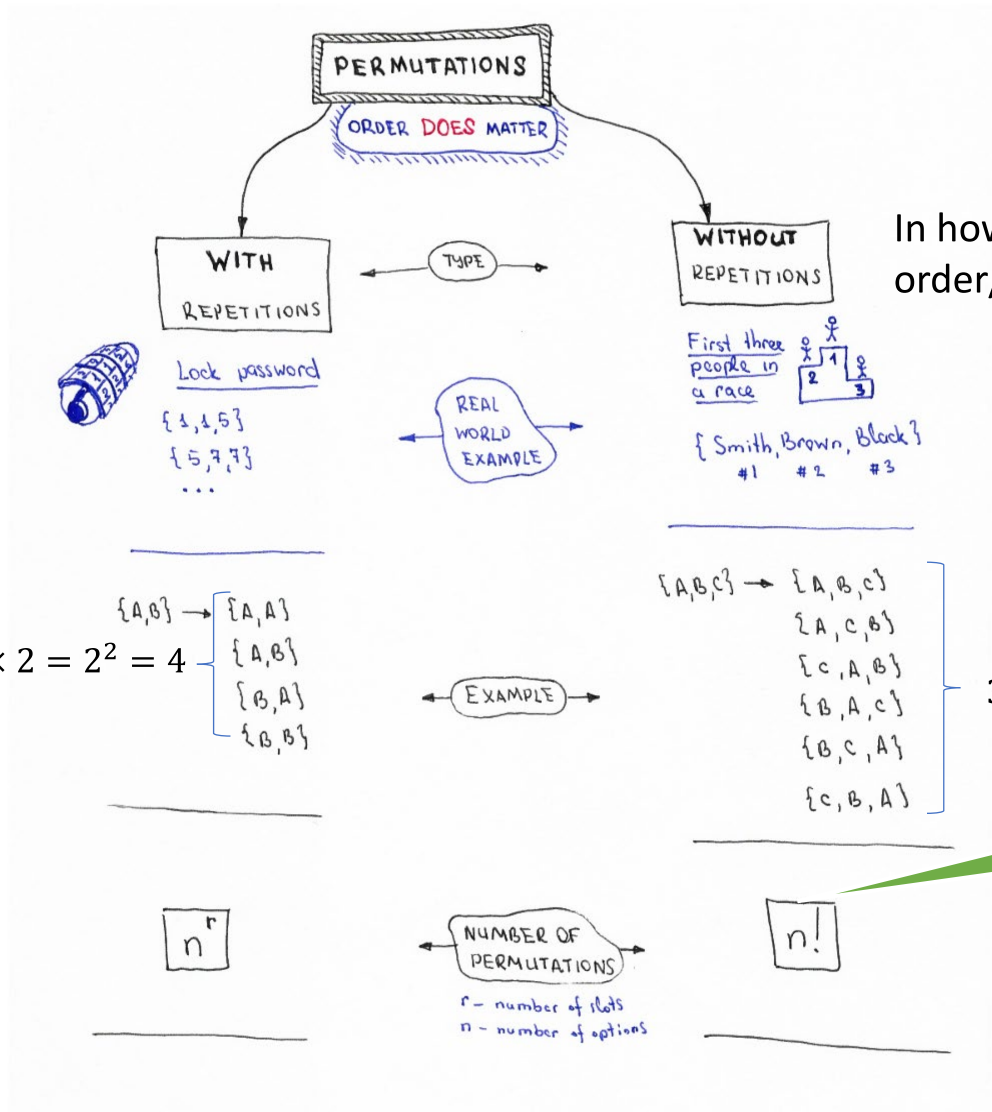
State Space Size Estimation

- Even if the used algorithm represents the state space using atomic states, we may know that internally they have a factored representation that can be used to estimate the problem size.
- The basic rule to calculate (estimate) the state space size for factored state representation with n fluents (variables) is:

$$|x_1| \times |x_2| \times \cdots \times |x_n|$$

where $|\cdot|$ is the number of possible values.

In how many ways can we order/arrange n objects?



$2 \times 2 = 2^2 = 4$

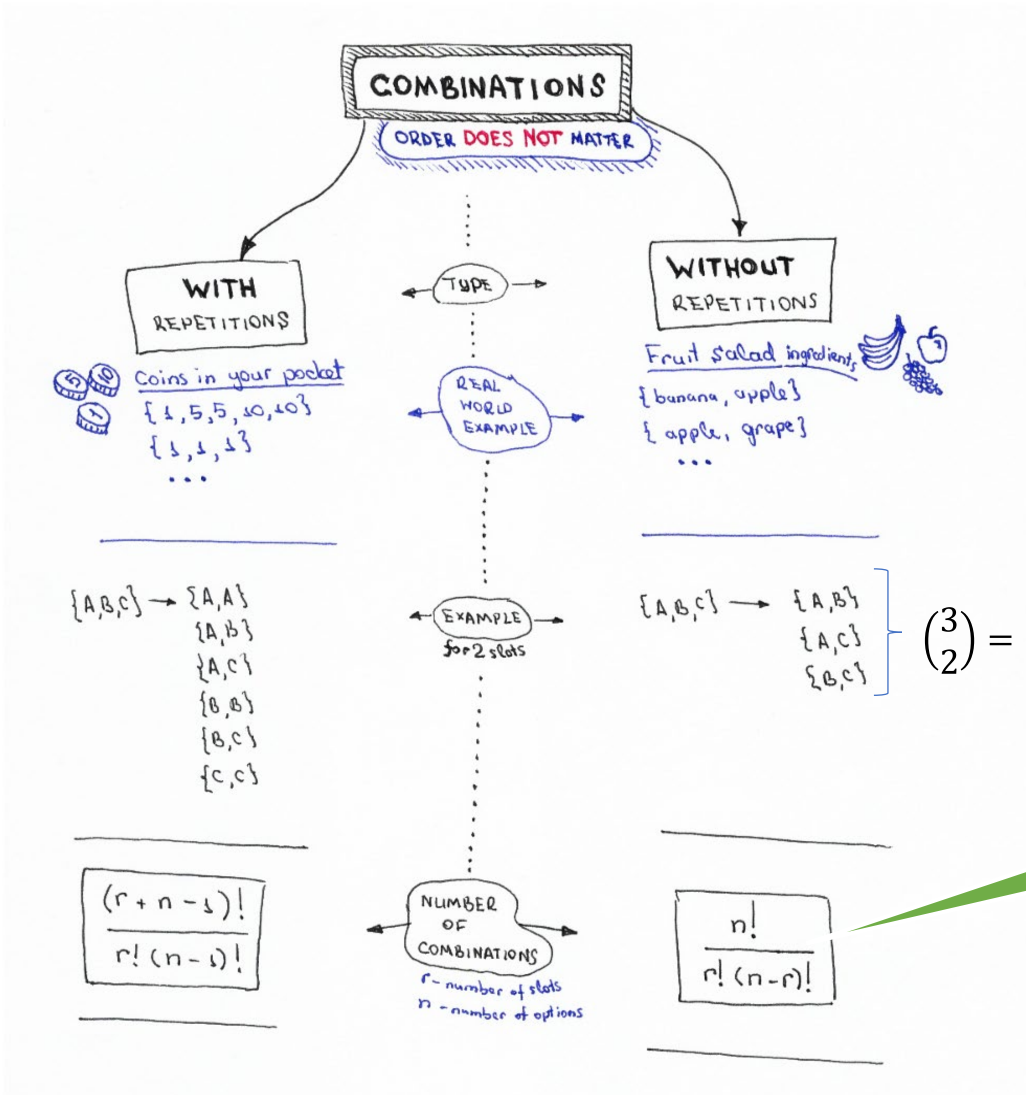
$3 \times 2 \times 1 = 6$

Factorial: $n! = n \times (n-1) \times \cdots \times 2 \times 1$

```
#Python
import math

print (math.factorial(23))
```

Source: Permutations/Combinations Cheat Sheets by Oleksii Trekhleb
https://itnext.io/permutations-combinations-algorithms-cheat-sheet-68c14879aba5



$$\binom{3}{2} = 3$$

Binomial Coefficient: $\binom{n}{r} = C(n,r) = {}_nC_r$

Read as "n choose r" because it is the number of ways can we choose r out of n objects?

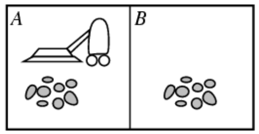
Special case for $r = 2$: $\binom{n}{2} = \frac{n(n-1)}{2}$

#Python
import scipy.special

 # the two give the same results
scipy.special.binom(10, 5)
scipy.special.comb(10, 5)

Source: Permutations/Combinations Cheat Sheets by Oleksii Trekhleb
https://itnext.io/permutations-combinations-algorithms-cheat-sheet-68c14879aba5

Example: What is the State Space Size?



Dirt

- **Permutation:** A and B are different rooms, order does matter!
- **With repetition:** Dirt can be in both rooms.
- There are 2 options (clean/dirty)

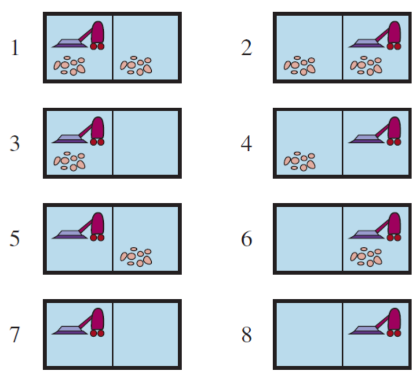
$\rightarrow 2^2$

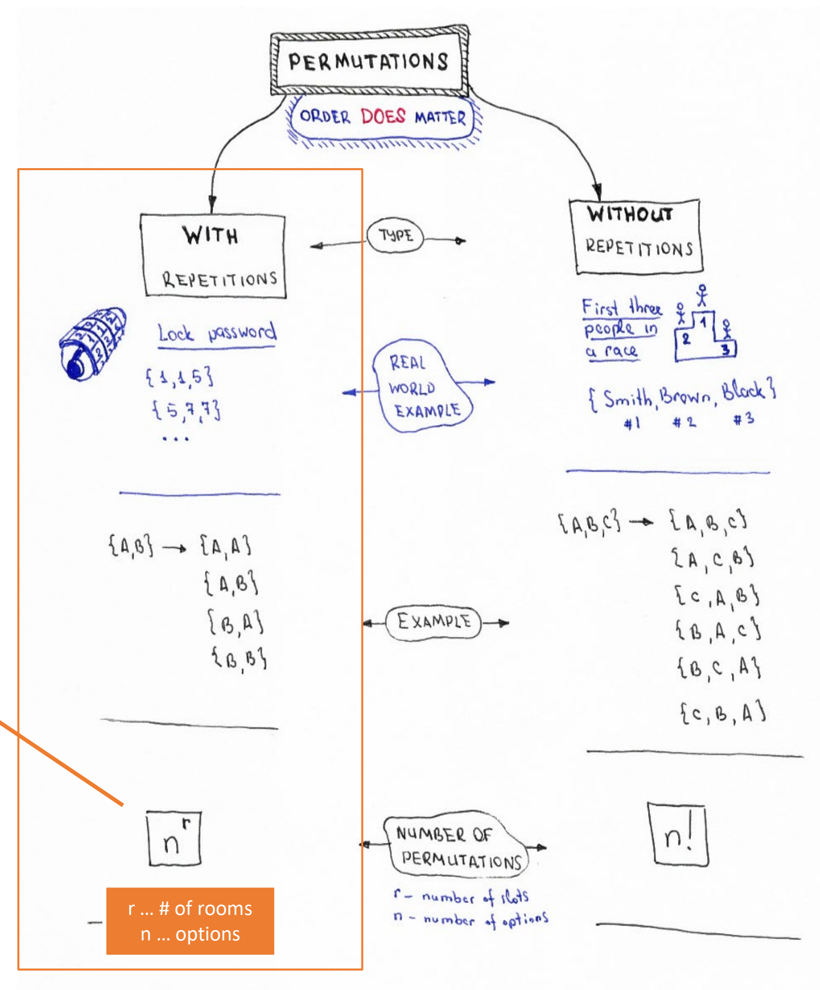
Robot location

- Can be in 1 out of 2 rooms.

$\rightarrow 2$

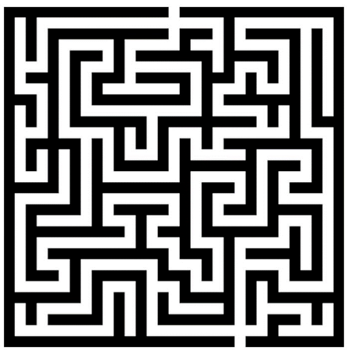
Total: $n = 2 \times 2^2 = 2^3 = 8$



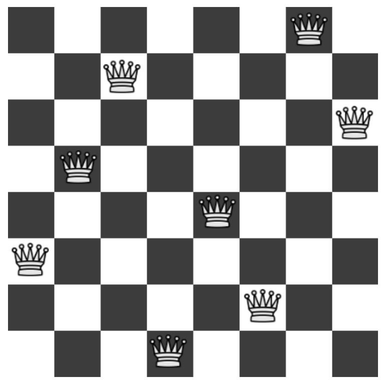


Examples: What is the State Space Size?

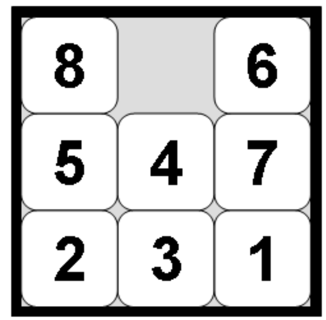
Often a rough upper limit is sufficient to determine how hard the search problem is.



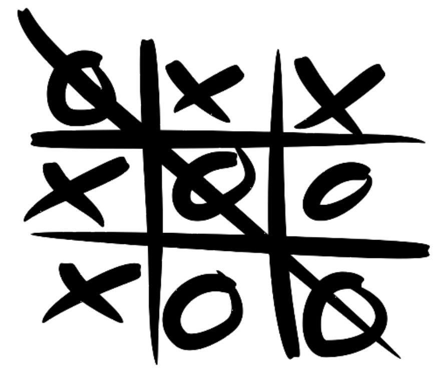
Maze



8-queens problem



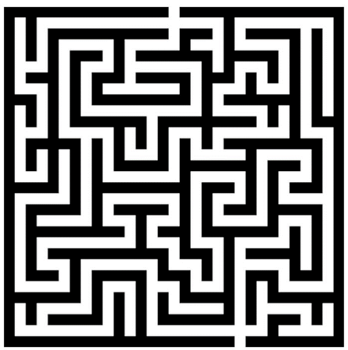
8-puzzle problem



Tic-tac-toe

Examples: What is the State Space Size?

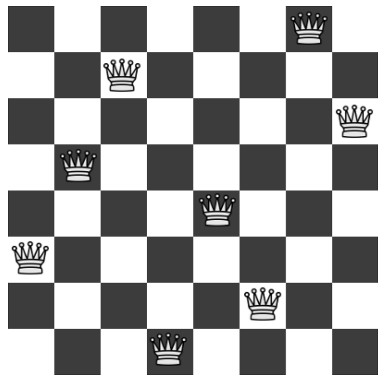
Often a rough upper limit is sufficient to determine how hard the search problem is.



Maze

Positions the agent can be in.

n = Number of white squares.



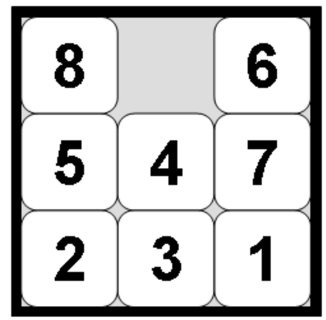
8-queens problem

All arrangements with 8 queens on the board.

$$n < 2^{64} \approx 1.8 \times 10^{19}$$

We can only have 8 queens:

$$n = \binom{64}{8} \approx 4.4 \times 10^{9}$$



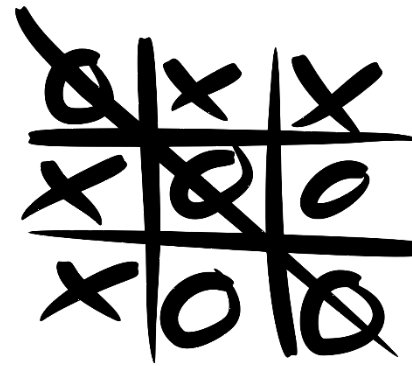
8-puzzle problem

All arrangements of 9 elements.

$$n \leq 9!$$

Half is unreachable:

$$n = \frac{9!}{2} = 181{,}440$$



Tic-tac-toe

All possible boards.

$$n < 3^{9} = 19{,}683$$

Many boards are not legal (e.g., all x's)

The actual number can be obtained by a depth-first traversal of the game tree.

Example: What is the Search Complexity?

- b: maximum branching factor = number of available actions?

  3

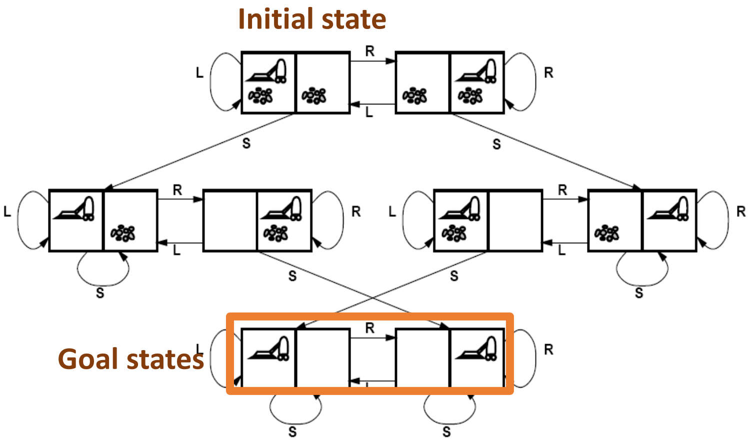
- m: the number of actions in any path? Without loops!

  4

- d: depth of the optimal solution?

  3

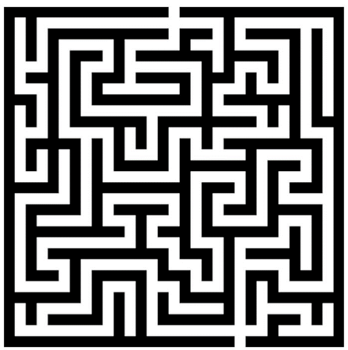
State Space with Transition Model



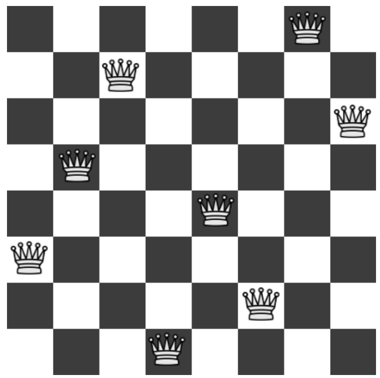
Examples: What is the Search Complexity?

b: maximum branching factor
m: max. depth of tree
d: depth of the optimal solution

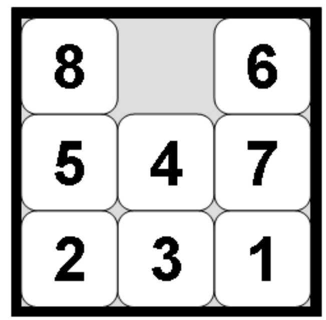
Often a rough upper limit is sufficient to determine how hard the search problem is.



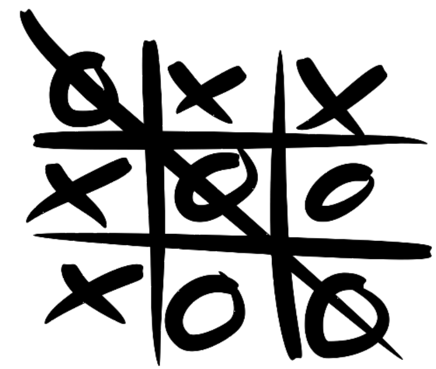
Maze



8-queens problem



8-puzzle problem



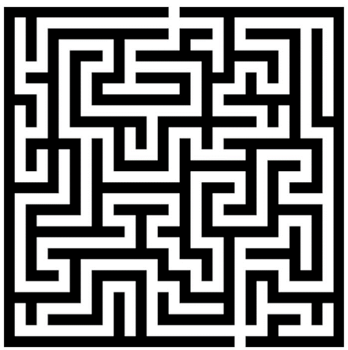
Tic-tac-toe

$b =$
$m =$
$d =$

Examples: What is the Search Complexity?

b: maximum branching factor
m: max. depth of tree
d: depth of the optimal solution

Often a rough upper limit is sufficient to determine how hard the search problem is.

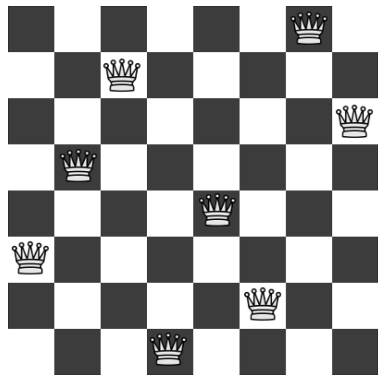


Maze

b = 4 actions

m = longest path to the goal or a dead end (bounded by $x \times y$)

d = shortest path to the goal (bounded by $x \times y$)

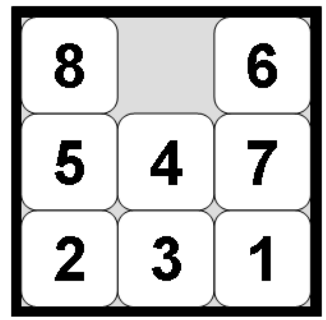


8-queens problem

b = ? What are the actions? Moving one Queen: $64 - 7 = 57$

m = We may have to try all: $\binom{64}{8} \approx 4.4 \times 10^9$

d = move each queen in the right spot = 8

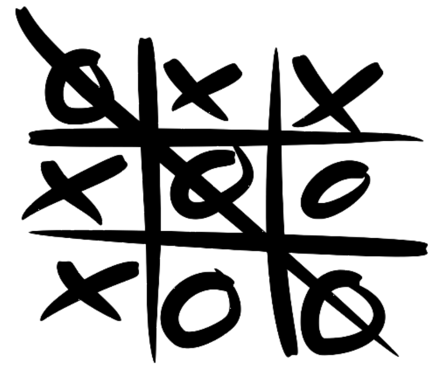


8-puzzle problem

b = 4 actions to move the empty tile.

m = Try all $O(9!)$

d = ???



Tic-tac-toe

b = 9 actions for the first move.
$m = 9$

d = 9 (if both play optimal)

Assignment

Q&A