Problem Solving as Search

CS161

Prof. Guy Van den Broeck

What is search?

- Basic AI problem
- Long history, active topic
- Basis for games, SAT, advanced reasoning
- Used in many other fields

- UCLA plays key role in history of search ☺

- Graduate CS class offered by Prof. Korf

What is search?

- Structure: Search 'engine'

- Sensitive to
  - problem formulation (today)
  - choice of strategy (next lectures)

Examples

- Puzzles
  - 8-puzzle
  - Missionaries and Cannibals
  - N-Queens
  - Crypto arithmetic
  - Rubik's cube
- Planning
  - Practical industry applications
    (e.g., Amazon delivery, Uber route and match, etc.)

How would **you** solve these?

Search Problem Formulation

- Common elements:
  - Initial state
  - Final/goal state or goal test/predicate
  - Actions
  - Transition model/successors
  - (Costs)
- <u>Task</u>: find sequence of actions
  to move from initial to goal state.
- Optimal solution has lowest cost

Search Problem Formulation

- Notions of state and action are not sharp!
- Design problem: art

- Example: 8-puzzle?

Observations

- States are atomic (no internal structure)
- States are discrete
- No percepts
- Deterministic transitions
- Choice of action space is important
  - Move tile x up, down, left, right?
  - Move blank up, down, left, right?

    More elegant, fewer actions!
- Actions may not be applicable to all states

Search Trees

- Example: 8-puzzle?

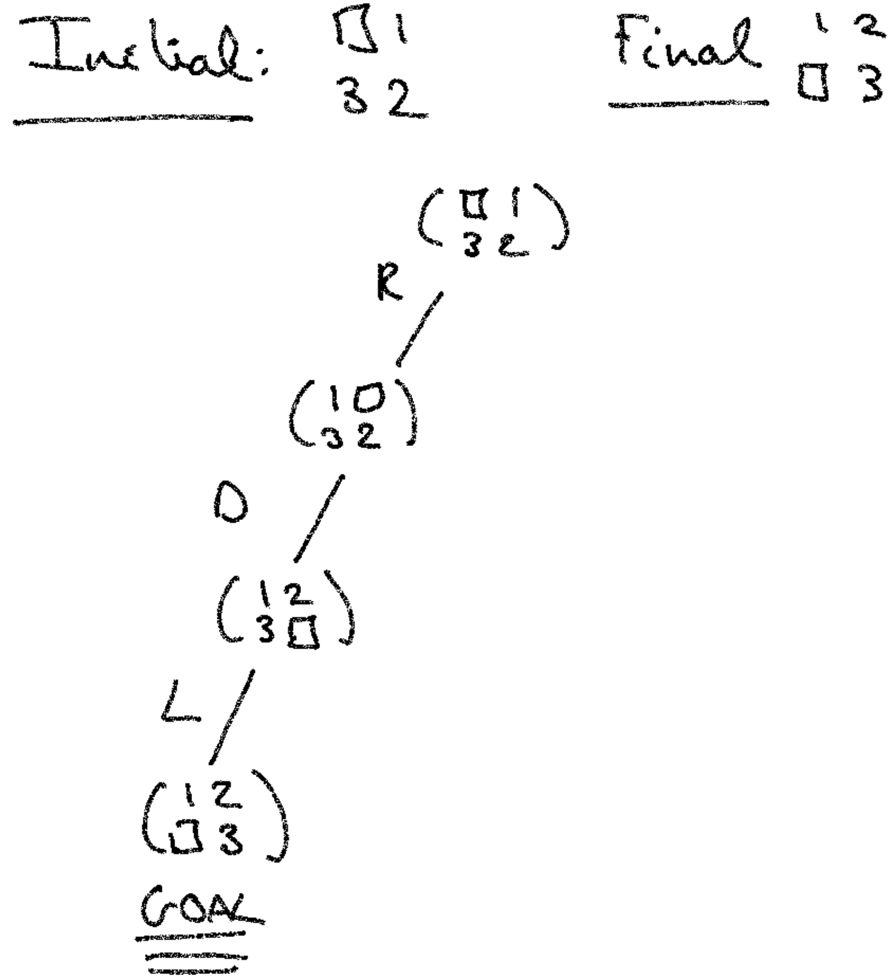
Observations

- Some paths include the goal state
- Solving the search problem is finding such path
- Some paths are infinite (tree infinite)
- Beware: Repeated states
- Search space vs. search tree
- Different solutions with different qualities

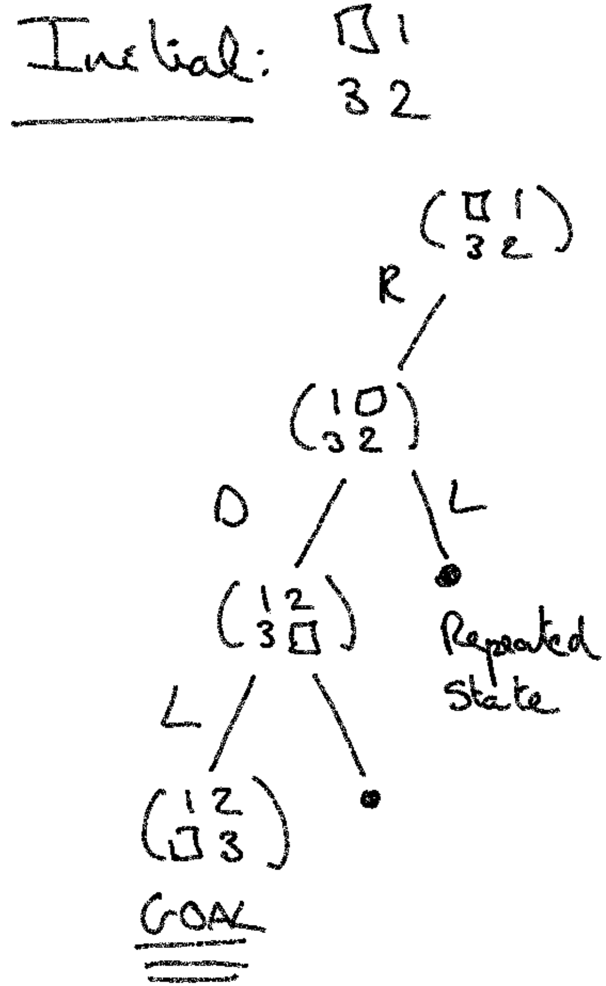
Search Trees: Finding Solutions

Initial: $\begin{matrix} \square & 1 \\ 3 & 2 \end{matrix}$ Final $\begin{matrix} 1 & 2 \\ \square & 3 \end{matrix}$

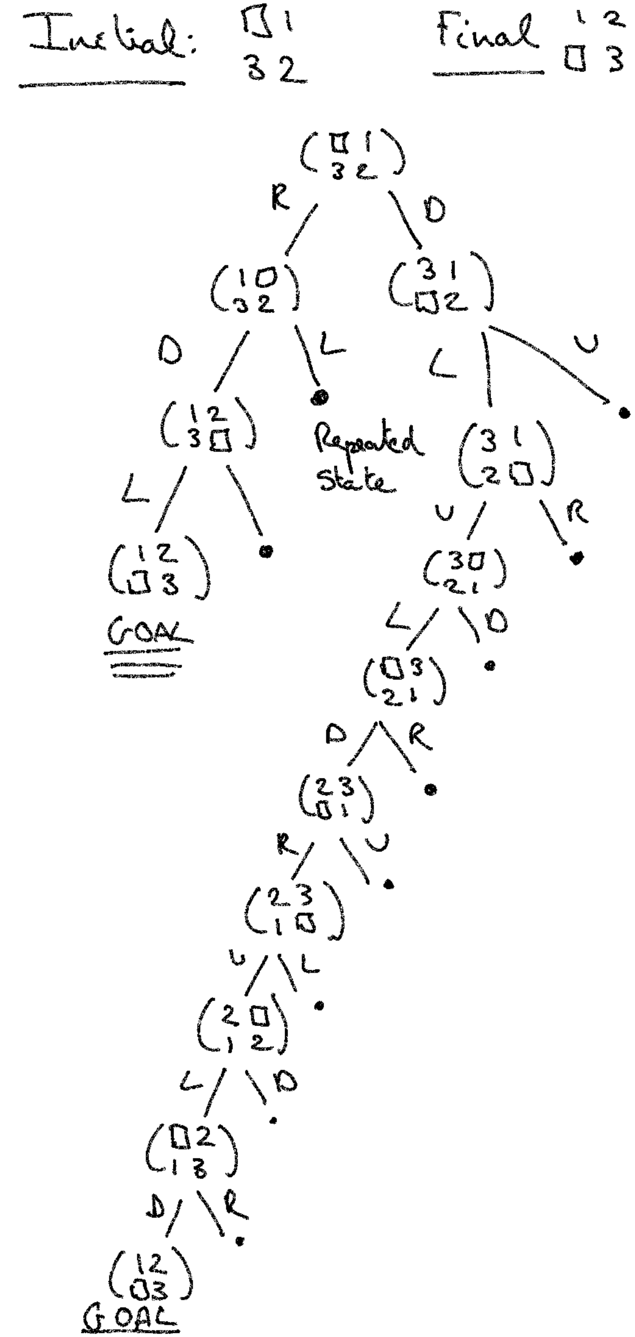
Search Trees: Finding Solutions



Search Trees: Finding Solutions



Search Trees: Finding Solutions



Observations

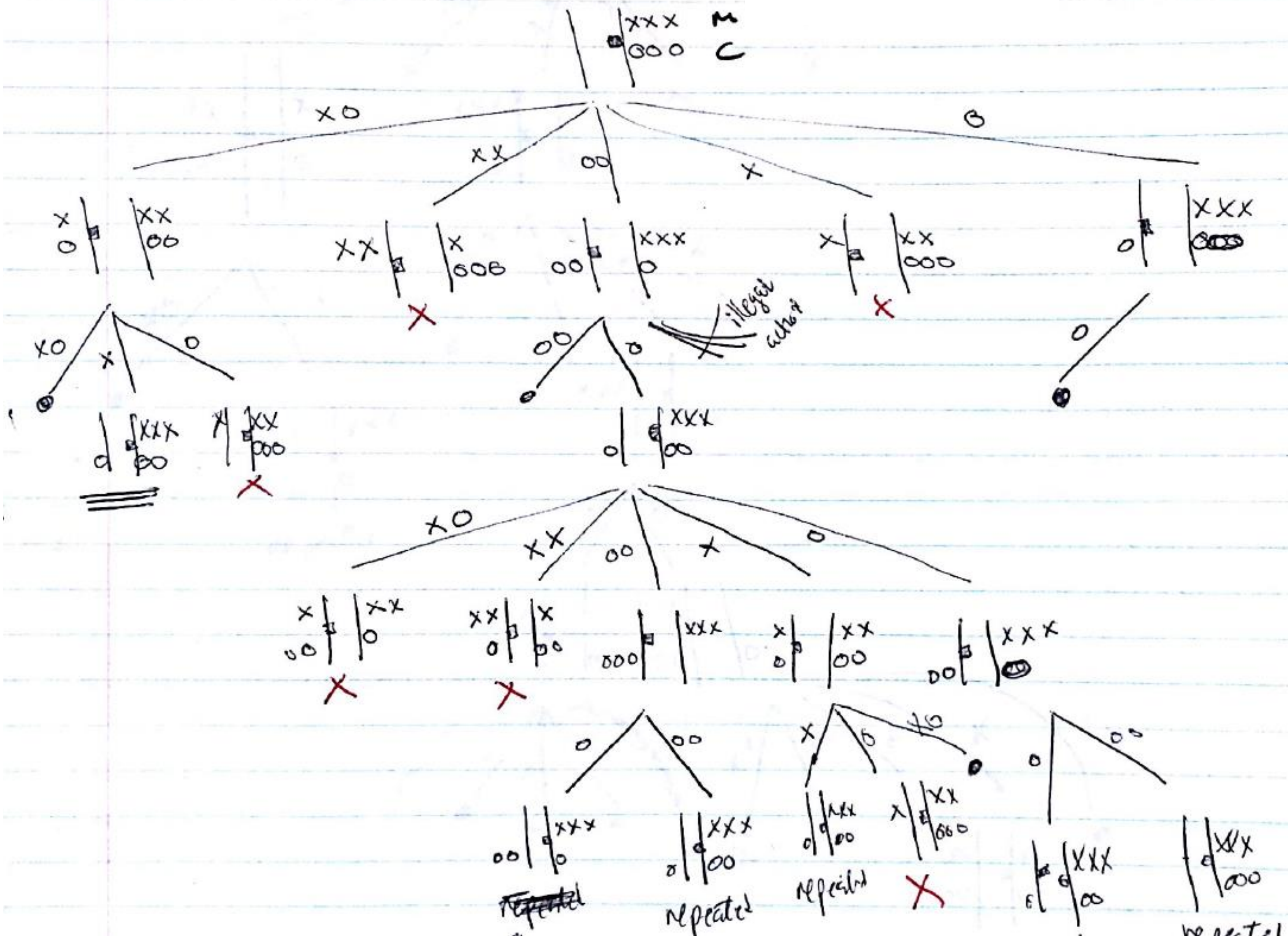
- Avoid repeated states or run into infinite loop
- Two solutions: cost 3 and cost 9

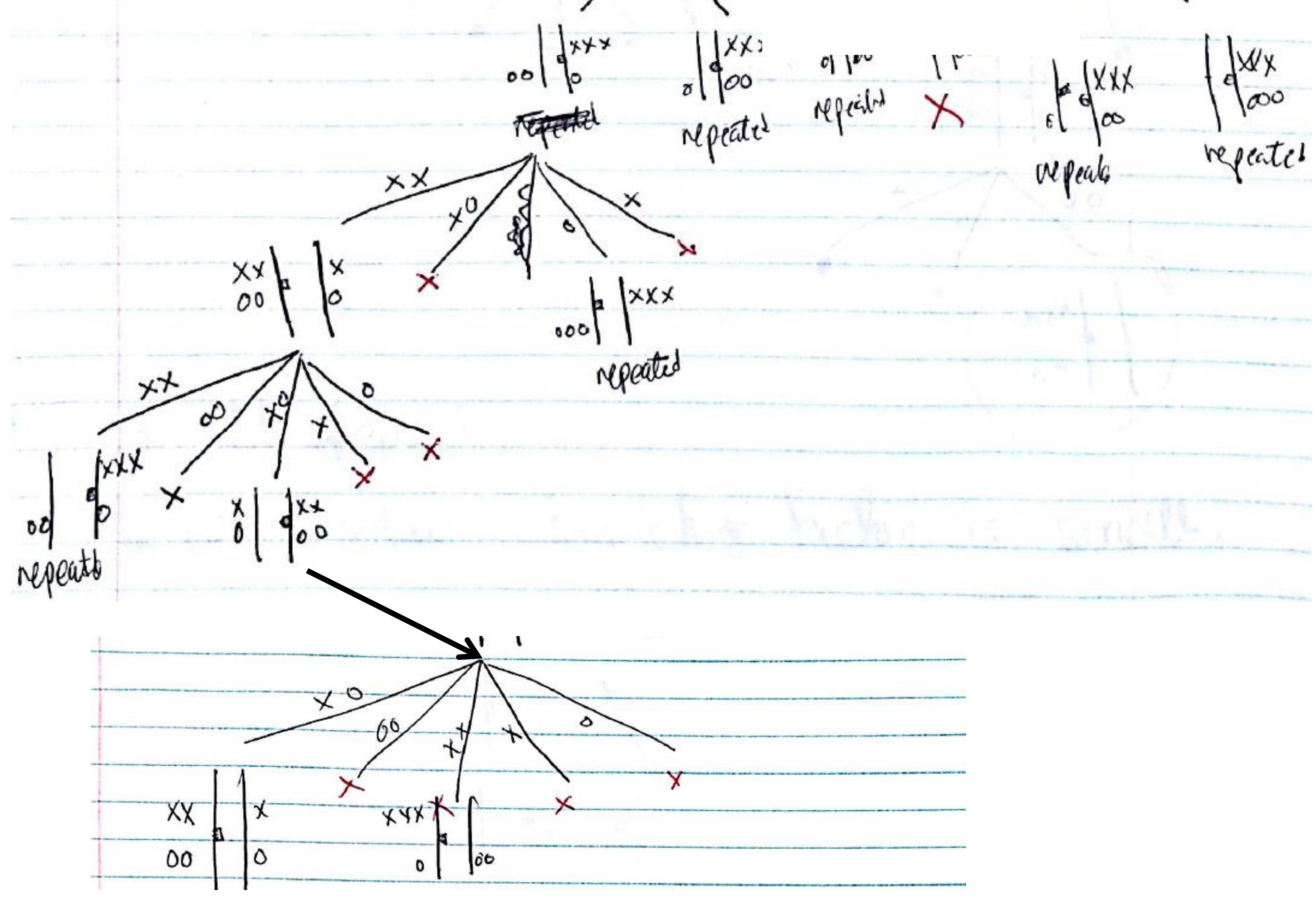
- Difficulty of problem: measured by parameters
  - Number of states
  - Branching factor
  - Solution depth
- How many unique states? 9! = 362,880

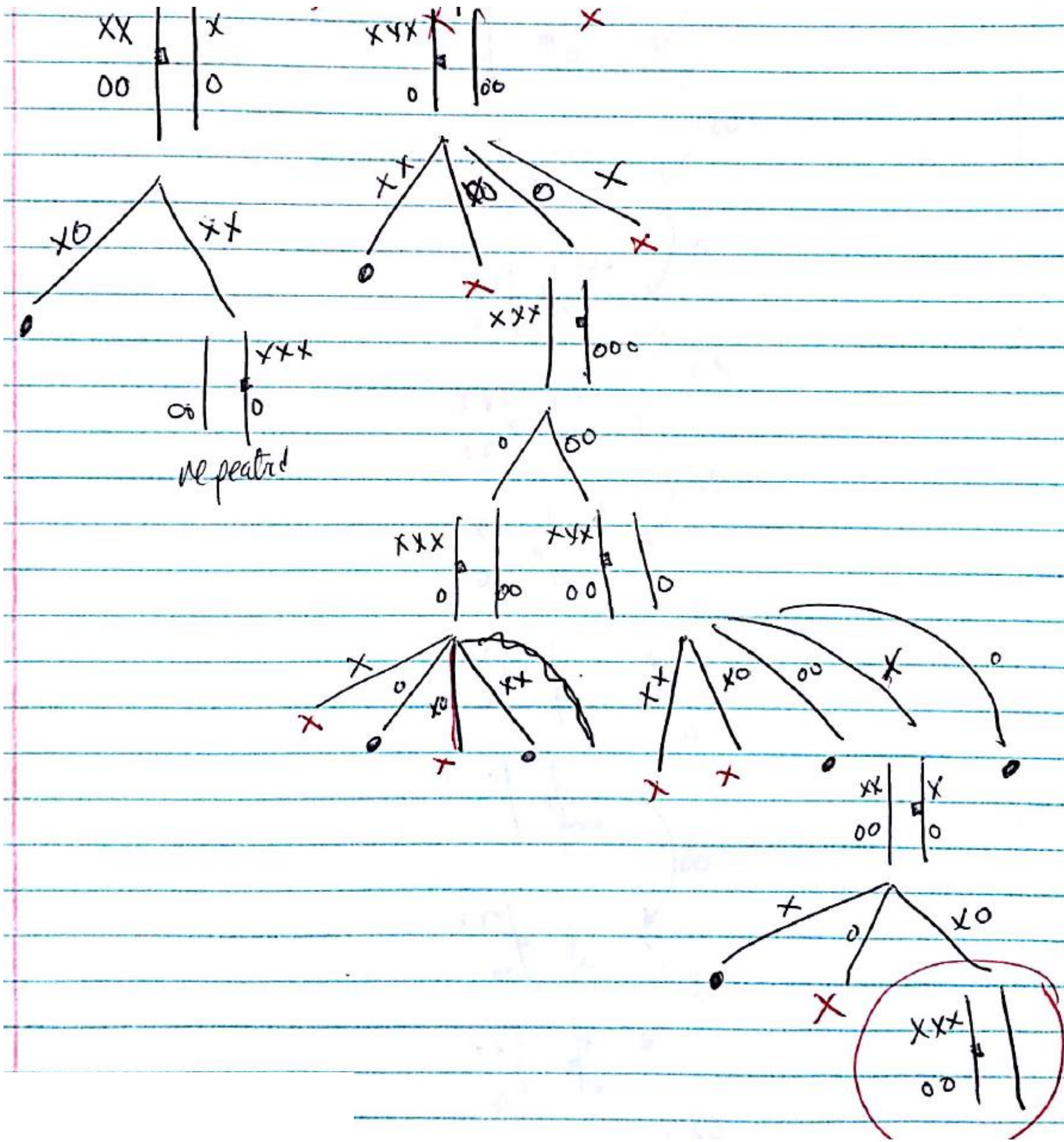
  16! = 20,922,789,888,000
  - 15-puzzle has trillions of states: few ms to solve
  - 24-puzzle has 10^25 states: one day to solve
- Solving sliding block puzzles is NP-complete!

Search Trees: Finding Solutions

- Example: Missionaries and Cannibals?


M
C
XO
XX
OO
X
O
illegal action
repeated
repeated
repeated

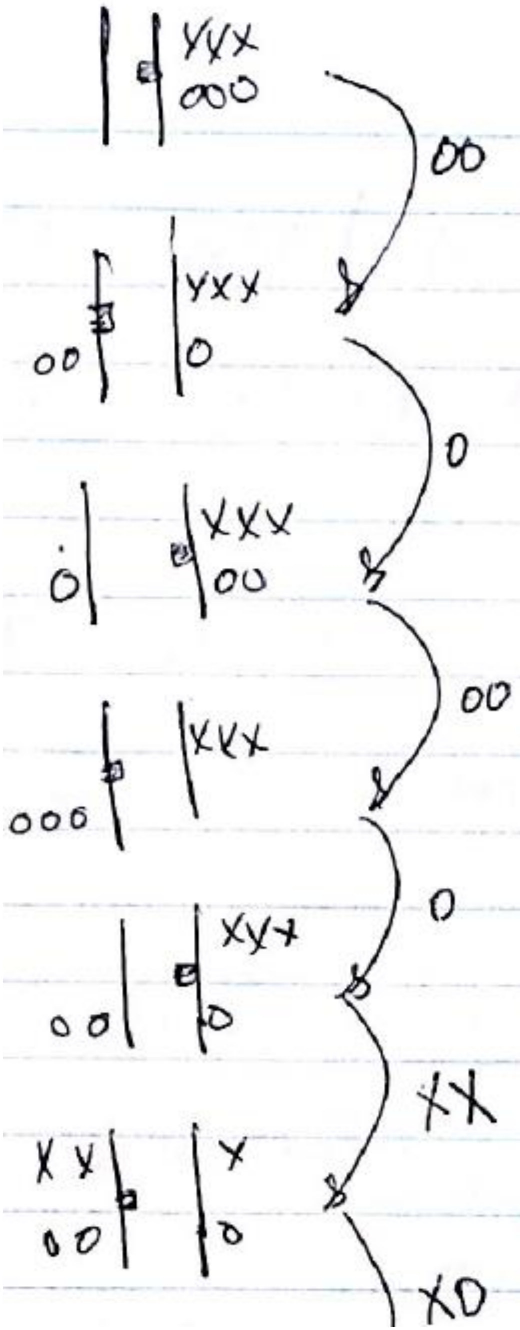

repeated
repeated
repeated
repeated
repeated
repeated

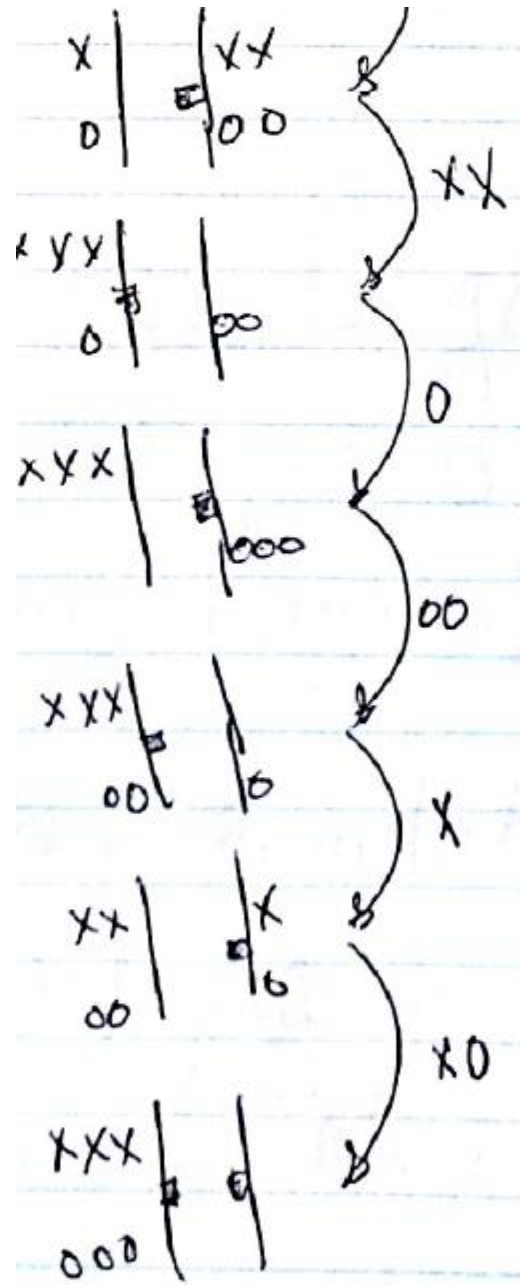

repeated

Observations

- Many choices for
  - State description
  - Actions
- Many correct, some better than others
- Find correct level of abstraction

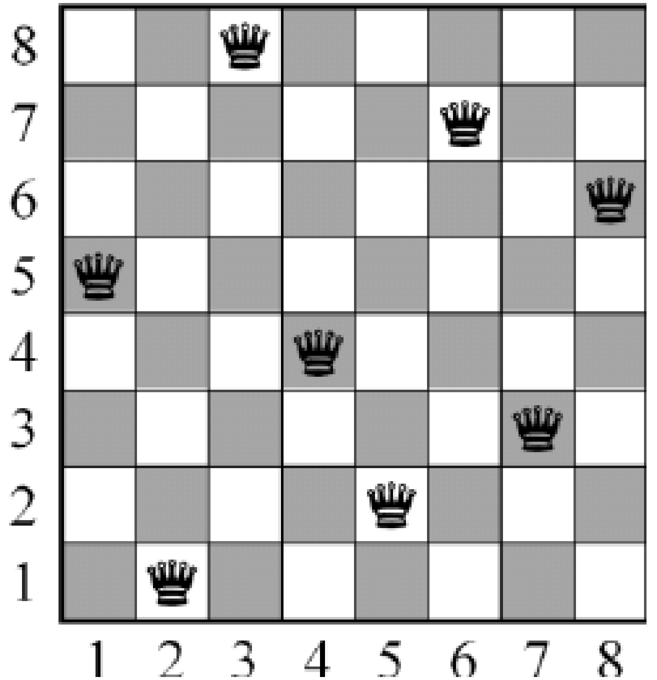
Solution





Search Trees: Finding Solutions

- Example: 8-Queens?



Observations

- State formulation:
  - Incremental state
  - Complete state
- May or may not allow bad intermediate states
- No goal state => goal predicate
- Solution cost:
  - Incremental state: 8
  - Complete state: 0-8 (Can be solved in 0 moves!)

Search Trees: Finding Solutions

- Example: Crypt-arithmetic?

FORTY	→	29786
TEN		850
TEN		850
+-------		+-------
SIXTY		31486

Observations

- Assign numbers to letters in fixed order

  (better: smallest number of remaining values)
- Incremental or complete state

Search Trees: Finding Solutions

- Example: Knut's Problem?
- Start: 4
- Actions
  - Factorial: x!
  - Sqrt: $\sqrt{x}$
  - Floor: $\lfloor x \rfloor$
- Transitions: math
- Goal: 5 => HOW?

Observations

- Conjecture:
  Any positive integer can be reached
- No bound on intermediate representation
- Infinite state space

- Example of class of problems: recursively defined objects

  Circuits, proofs, programs, etc.

More Real-World Search Problems

- Route finding

  Amazon, computer networks, travel planning, airline planning, navigation systems
- Traveling salesperson problems

  Planning circuit board drill movements, mail delivery, street cleaning, etc.
  What is a state?
- Hardware layout
- Continuous robot navigation
- Scheduling
- Theorem proving

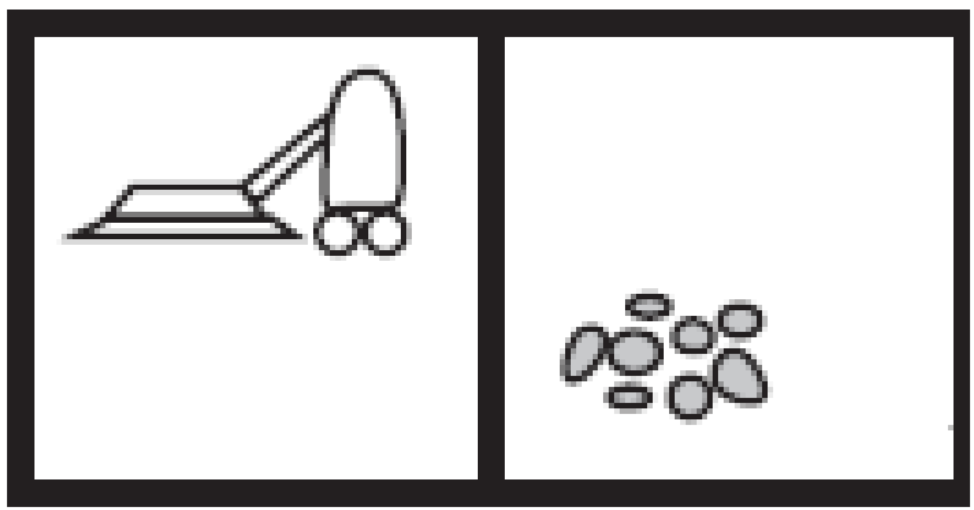
Extensions/Complications

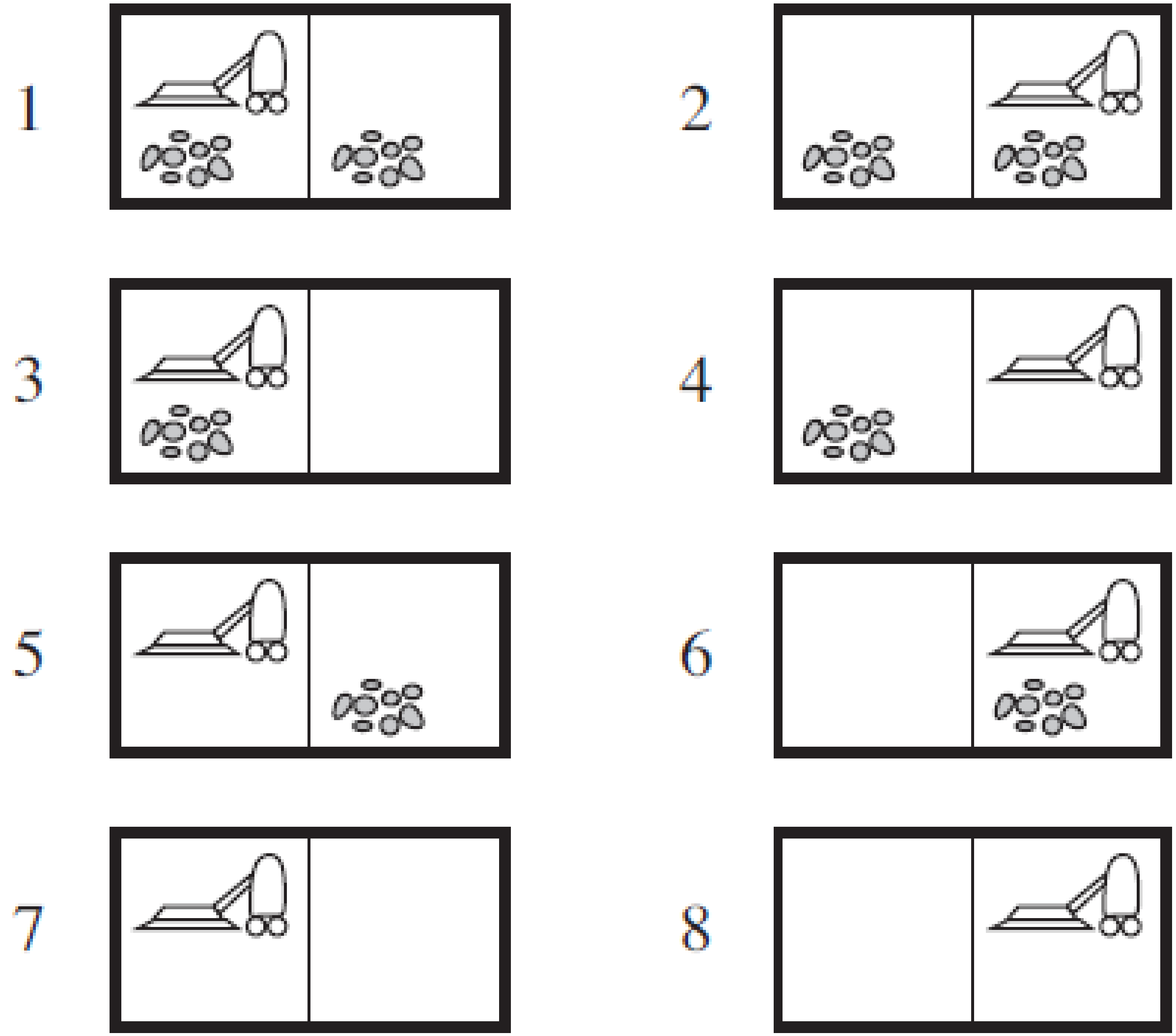
- Minimizing solution cost (‘optimal solution’)
  - Cost of solution is sum of action costs
- Note sure about effect of actions
  - Non-determinism (see probability later)
- Note sure about initial state

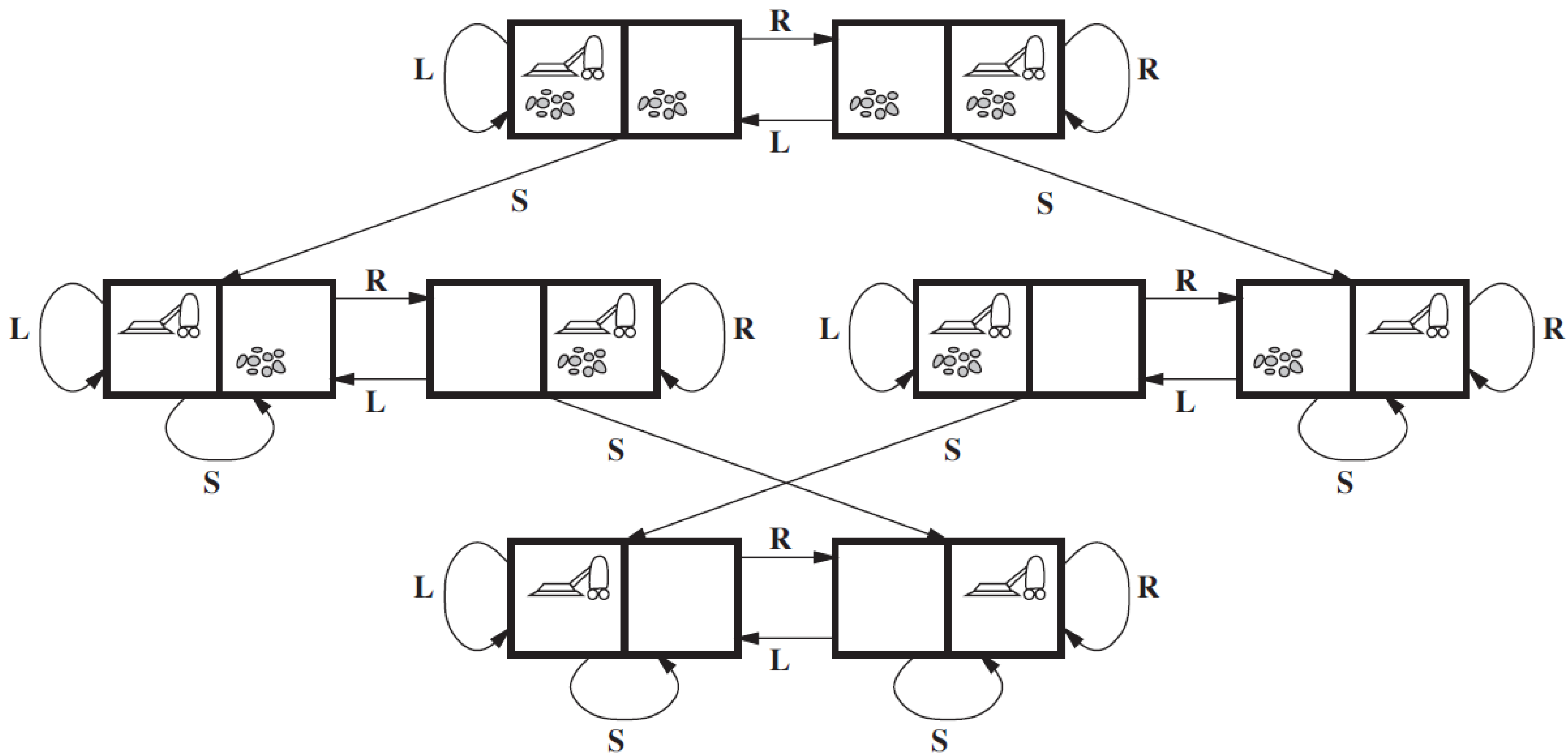
- Contingent planning: action based on sensing
- Conformant planning: plan without sensors

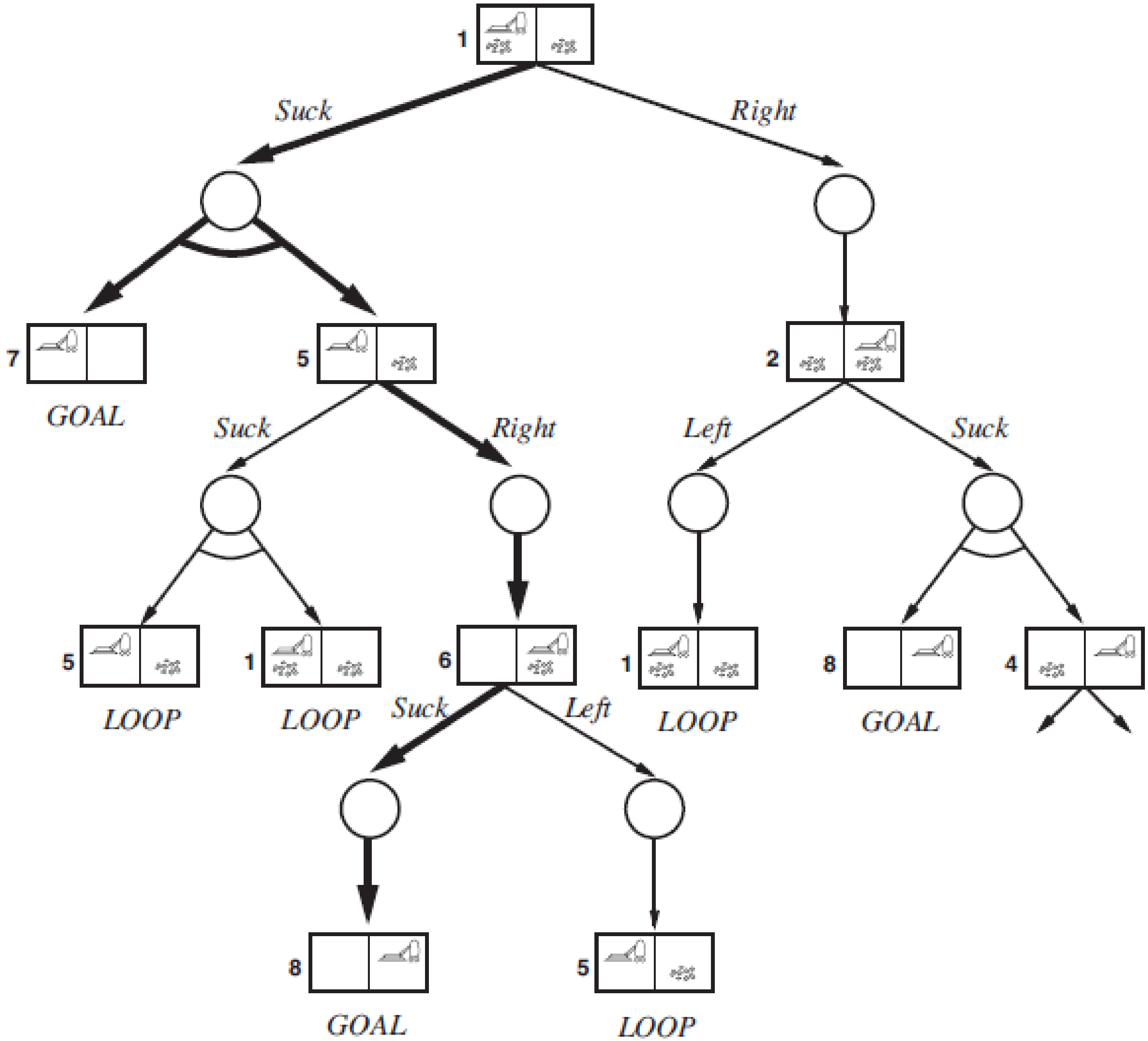
  Example: Not knowing initial state (see next)

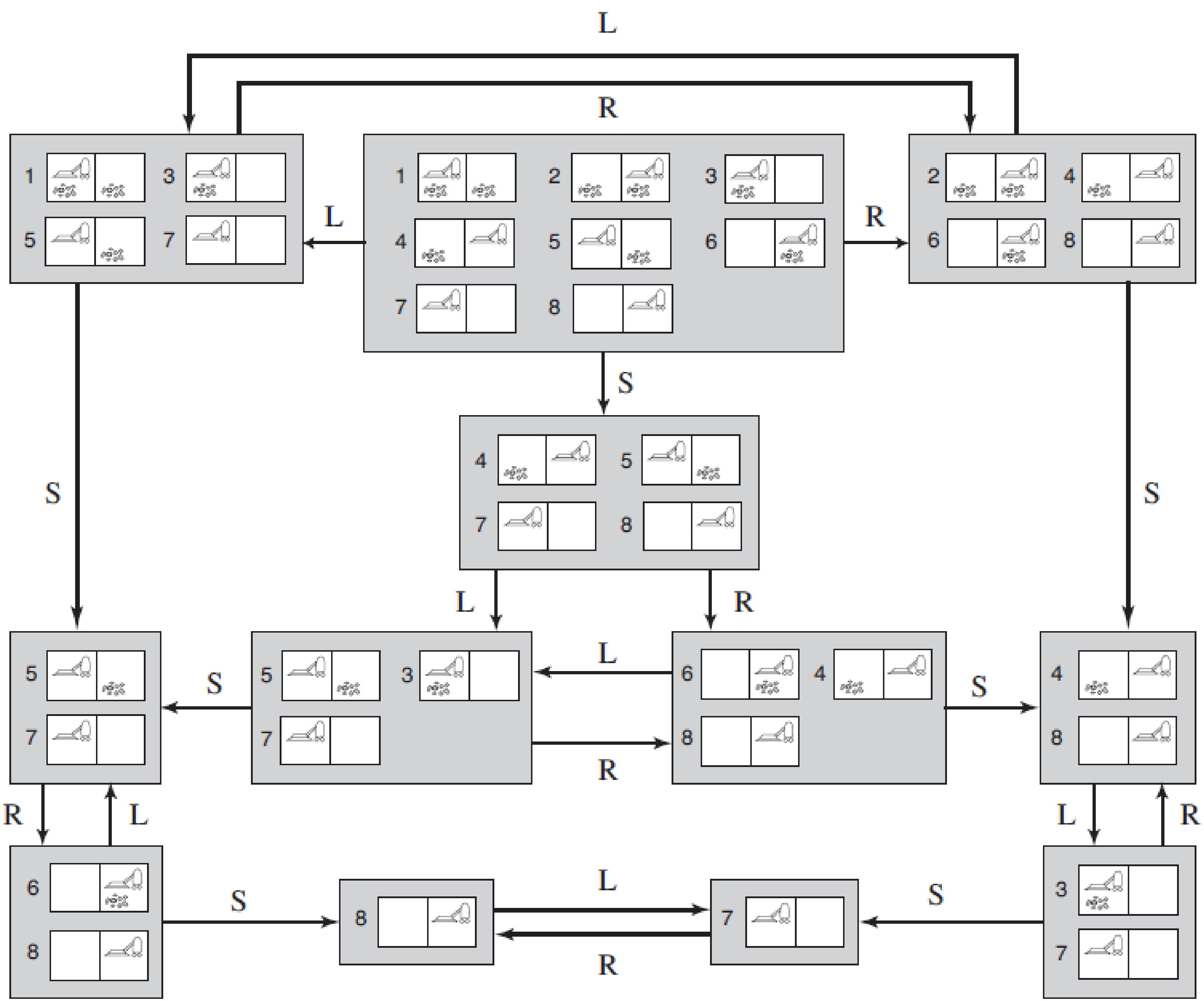
Vacuum Cleaner World




1
2
3
4
5
6
7
8


R
L
R
L
S
S
R
L
R
L
S
R
L
R
L
S
S
S
S
R
L
R
L
S
S


1
Suck
Right
7
GOAL
5
Suck
Right
2
Left
Suck
5
LOOP
1
LOOP
6
Suck
Left
1
LOOP
8
GOAL
4
8
GOAL
5
LOOP


L
R
L
R
S
S
S
L
R
L
S
S
R
R
L
L
R
S
L
S
R

Observations

- When uncertain about initial state,
  use belief states.
- Belief state is a set of deterministic states

- See later: assign probability to states