Knowledge-based Agents

Agents which use logic to represent domain knowledge and inference to reason about that knowledge.

One way to solve a problem…

1. Devise an algorithm to solve the problem
2. Write down algorithm in a programming language
3. Execute the program

Another way to solve a problem…

1. Identify the knowledge needed to solve the problem
2. Write down knowledge in a knowledge representation language
3. Use logical consequences of knowledge to solve problem

Contrast

- Procedural knowledge
  - “how to” knowledge
  - e.g. programs
  - languages: C, C++, Java, ...
- Declarative knowledge
  - descriptive knowledge
  - e.g. databases
  - languages: propositional logic, first-order predicate logic, logic programming languages, production systems, relational databases & SQL, ...

Example: credit card authorization

Handle majority of transactions autonomously.

Pass exceptional cases (e.g., unusual travel patterns) to a human along with recommendation and reasons.

Charge Card Authorization (American Express)

- Real time kb system; about 1000 rules
- About 2/3 of transactions are so routine the system handles them autonomously
- If it finds anything extraordinary, the case is given to a person for a final decision
  - example: unusual travel patterns
- System is able to show the authorizer its reasoning, aiding in a decision

Example: mortgage authorization

One of the largest mortgage companies in U.S.

Refer hard cases to human underwriter

Mortgage Underwriting (Countrywide Funding)

- (Was) one of the largest mortgage companies in U.S.
- Increasing number of loans, shortage of underwriters
- Solution: knowledge-based system
  - about 1000 rules
  - processes over 8500 loans per month in 300 branches
  - refers hard cases to human underwriter
- Benefits:
  - estimated 0.9 million saved annually and rising
  - consistency, removal of human bias
  - training tool

Formal logic

- A logic consists of
  - syntax: What is an acceptable sentence in the language?
  - semantics: What do the symbols and sentences in the language mean?
  - proof procedure: How do we construct valid proofs?

Syntax of propositional logic

- Alphabet
  - propositional symbols (Boolean variables): *p*, *q*, *r*, ...,
  - two constant symbols: **true**, **false**
  - brackets: (,)
  - propositional connectives: $\neg$, $\wedge$, $\vee$, $\Rightarrow$, $\Leftrightarrow$
- Syntax
  - a propositional symbol is a sentence (formula)
  - if α and β are sentences (formulas) so are:

    $\neg\alpha$, (α), $\alpha \wedge \beta$, $\alpha \vee \beta$, $\alpha \Rightarrow \beta$, $\alpha \Leftrightarrow \beta$

Syntax of predicate logic

- Alphabet
  - constants, variables, predicates, functions
  - connectives: $\neg$, $\wedge$, $\vee$, $\Rightarrow$, $\Leftrightarrow$
  - quantifiers: $\forall$, $\exists$
- Syntax
  - A term is a constant, a variable, or f(t_1, …, t_n), where t_1, …, t_n are terms and f is a function symbol
  - P(t_1, …, t_n) is a sentence, where t_1, …, t_n are terms and P is a predicate
  - if α and β are sentences so are:

    $\neg\alpha$, (α), $\alpha \wedge \beta$, $\alpha \vee \beta$, $\alpha \Rightarrow \beta$, $\alpha \Leftrightarrow \beta$
  - if α is a sentence and x is a variable then the following are also sentences:

    $\forall x \cdot \alpha$, $\exists x \cdot \alpha$

Logical implication (deduction)

A set of sentences Γ logically implies a sentence ϕ iff every interpretation that satisfies Γ also satisfies ϕ

We write this as,

$$\Gamma \models \phi$$

Proofs in logic

- Many proof procedures
  - transformational proofs
  - natural deduction
  - resolution refutation
- We write,

$$\Gamma \vdash \phi$$

to mean there is a proof of ϕ from Γ

Suppose we have the propositions:

a = The car has gas.
b = I can go to the store.
c = I have money.
d = I can buy food.
e = The sun is shining.
f = I have an umbrella.
g = I can go on a picnic.

Write axioms to represent the statements:

1. If the car has gas, then I can go to the store.
2. I can buy food if I can go to the store and I have money.
3. If I can buy food and either the sun is shining or I have an umbrella, I can go on a picnic.
4. The car has gas.
5. I have money.
6. The sun is not shining.

Write axioms to represent the statements:

1. If the car has gas, then I can go to the store.

   $a \Rightarrow b$

2. I can buy food if I can go to the store and I have money.

   $(b \wedge c) \Rightarrow d$

3. If I can buy food and either the sun is shining or I have an umbrella, I can go on a picnic.

   $(d \wedge (e \vee f)) \Rightarrow g$

4. The car has gas.

   a

5. I have money.

   c

6. The sun is not shining.

   $\neg e$

Example queries

- I can buy food?

  $\Gamma \models d$?

- If I have an umbrella, I can go on a picnic?

  $\Gamma \models (f \Rightarrow g)$?

- If I can go on a picnic, then I don't have an umbrella?

  $\Gamma \models (g \Rightarrow \neg f)$?

Holmes scenario

Mr. Holmes lives in a high crime area and therefore has installed a burglar alarm. He relies on his neighbors to phone him when they hear the alarm sound. Mr. Holmes has two neighbors, Dr. Watson and Mrs. Gibbon.

Unfortunately, his neighbors are not entirely reliable. Dr. Watson is known to be a tasteless practical joker and Mrs. Gibbon, while more reliable in general, has occasional drinking problems.

Mr. Holmes also knows from reading the instruction manual of his alarm system that the device is sensitive to earthquakes and can be triggered by one accidentally. He realizes that if an earthquake has occurred, it would surely be on the radio news.

Logic-based representation

Suppose we have the propositions:

w	=	Watson calls saying the alarm is going.
g	=	Gibbon calls saying the alarm is going.
a	=	The alarm is going.
b	=	There is burglary in progress.
r	=	There is a radio news report about an earthquake.
e	=	There is an earthquake happening.

Planning with Certainty

Goal-based agents that decide what to do by finding sequences of actions that lead to desirable states.

Planning

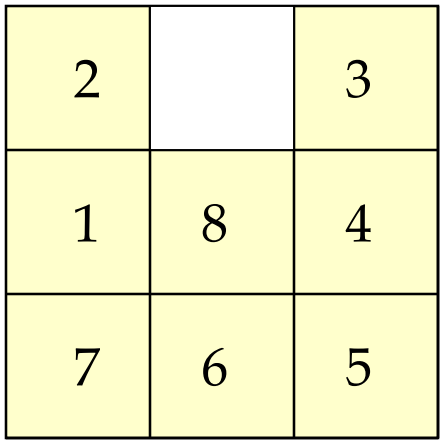
- Planning:
  - Given: an agent's ability (actions it can perform), its goal, and the current state of the world
  - Find: a sequence of actions to achieve a goal
- Initial assumptions:
  - Agent's actions are deterministic
  - No exogenous events beyond the control of the agent
  - World is fully observable
  - Time progresses discretely

Actions

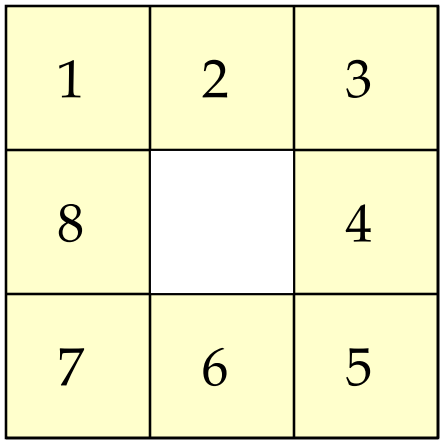
- A deterministic *action* is a partial function from states to states
- STRIPS/PDDL representation
  - *preconditions* of an action specify set of conditions that must be true for action to have desired effect
  - *effects* of an action specifies what changes as a result of performing the action:
    - set of conditions that become true (add list)
    - set of conditions that become false (delete list)

Example: 8-puzzle

Initial configuration



Goal configuration



Using predicates to represent states for 8-puzzle

On(x, i, j)	tile x is on cell i, j
Clear(i, j)	cell i, j is clear (empty)
Adj(i, j, k, l)	cell i, j is adjacent to cell k, l

State

2		3
1	8	4
7	6	5

Representation

On(2,1,1), Clear(1,2), On(3,1,3)

On(1,2,1), On(8,2,2), On(4,2,3)

On(7,3,1), On(6,3,2), On(5,3,3)

Adj(1,1,1,2), Adj(1,1,2,1), ...

STRIPS/PDDL action for 8-puzzle

Move(x, i, j, k, l)	move tile x from cell i, j to cell k, l
preconditions:	On(x, i, j), Clear(k, l), Adj(i, j, k, l)
effects:	On(x, k, l), Clear(i, j),
	¬On(x, i, j), ¬Clear(k, l)

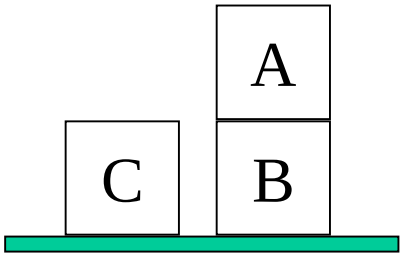
Updating rule

$$S_{i+1} \leftarrow (S_i - \text{delete}(\text{action})) \cup \text{add}(\text{action})$$

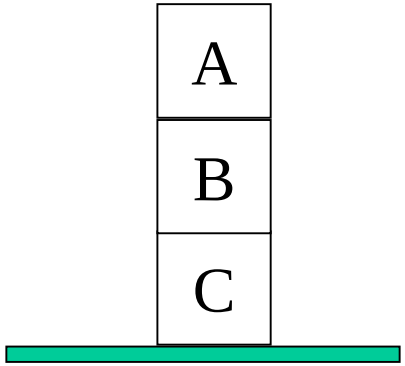
where $\text{precondition}(\text{action}) \subseteq S_i$

Example: Blocks world

Initial configuration



Goal configuration

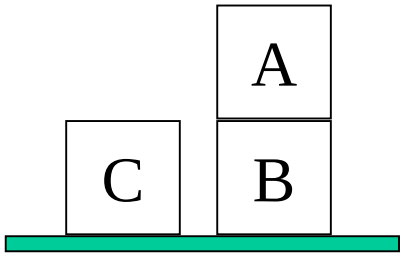


Actions: Stack, Unstack, Move

Using predicates to represent states for blocks

Clear(*x*)	*x* is clear
On(*x*, *y*)	*x* is directly on *y*
OnTable(*x*)	*x* is on the table

State



Representation of state

Clear(C), Clear(A),

On(A, B),

OnTable(C), OnTable(B)

STRIPS actions for blocks world

Move(x, y, z)
- preconditions: On(x, y), Clear(x), Clear(z), $x \neq z$
- effects: On(x, z), Clear(y), ¬On(x, y), ¬Clear(z)

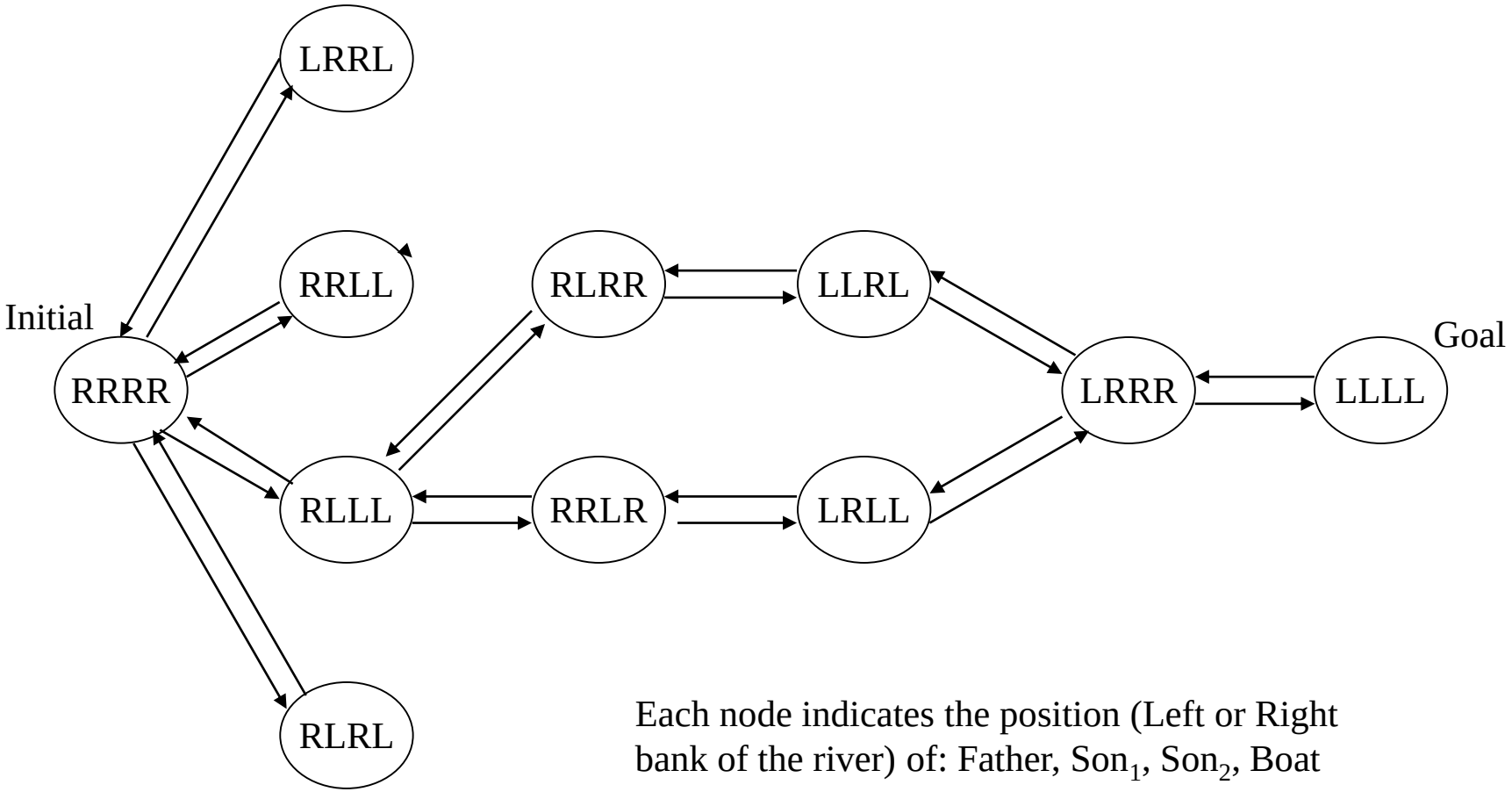
Stack(x, y)
- preconditions: OnTable(x), Clear(x), Clear(y), $x \neq y$
- effects : On(x, y), ¬OnTable(x), ¬Clear(y)

UnStack(x, y)
- preconditions: On(x, y), Clear(x)
- effects : OnTable(x), Clear(y), ¬On(x, y)

Example: river crossing puzzle

A father, his two sons, and a boat are on one side of a river. The capacity of boat is 100 kg. The father weighs 100 kg and each son weighs 50 kg. How can they get across the river?

Search graph for River Crossing Puzzle



Each node indicates the position (Left or Right bank of the river) of: Father, Son_1, Son_2, Boat

Solving planning problems using forward search

- Heuristic evaluation function:
  - Want: search guided by a heuristic function that is automatically extracted
  - Idea: relax a problem *P* to a simpler problem *P*', which can be solved efficiently. Use the solution to the simpler problem *P*' as an estimate for original problem *P*
  - Here: Relax the planning problem by simply ignoring delete effects
    - actions only add new atoms to the state, never remove any
    - problem is solved as soon as each subgoal has been added by some action
    - length of an optimal relaxed solution is an admissible heuristic

Solving planning problems using forward search

- Search strategy:
  - Use A* and admissible heuristic to find optimal plan
  - Use local search and heuristic to find good plans
    - give up on optimality
    - use heuristic to choose best neighbour to move to