ARTIFICIAL INTELLIGENCE

The Very Idea

Vasant G. Honavar
Dorothy Foehr Huck and J. Lloyd Huck Chair in Biomedical Data Sciences and Artificial Intelligence
Professor of Data Sciences, Informatics, Computer Science, Bioinformatics & Genomics and Neuroscience
Director, Artificial Intelligence Research Laboratory
Director, Center for Artificial Intelligence Foundations and Scientific Applications
Associate Director, Institute for Computational and Data Sciences
Pennsylvania State University

vhonavar@psu.edu
http://faculty.ist.psu.edu/vhonavar
http://ailab.ist.psu.edu

All representations are wrong, but some are useful
A lesson from Alice in Wonderland

- “What a useful thing a pocket-map is!” I remarked.
- “Map making. That's another thing we've learned from your Nation”, said Mein Herr. “But we've carried it much further than you.
- “What do you consider the largest map that would be really useful?''
- About six inches to the mile.
- Only six inches! exclaimed Mein Herr.
- “We very soon got to six yards to the mile. Then we tried a hundred yards to the mile. And then came the grandest idea of all ! We actually made a map of the country, on the scale of a mile to the mile!''

Sylvie and Bruno Concluded, Lewis Carroll, 1893

All representations are wrong, but some are useful
A lesson from Alice in Wonderland

- “Have you used it much?” I enquired.
- “It has never been spread out, yet," said Mein Herr
- “The farmers objected: they said it would cover the whole country, and shut out the sunlight! So we now use the country itself, as its own map, and I assure you it does nearly as well.”
- Lesson: A representation must be a cartoon of reality to be useful
  - It must retain details relevant to the problem at hand
  - It must discard details that have little bearing on the problem at hand

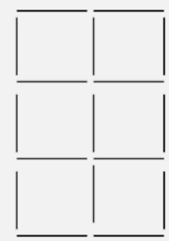
Sylvie and Bruno Concluded, Lewis Carroll, 1893

All representations are wrong, but some are useful

- The most accurate representation of the world is the world itself
- Representations are cartoons
  - They keep some details, throw out other details
  - The choice of what to discard is as important as that of what to keep
  - More detail is not necessarily better
- Because all representations are cartoons of reality, it is inevitable that they are in some respects wrong
- All reasoning about the natural world relies on representations
- All reasoning about the natural world, by humans or machines, will at some point yield conclusions that are incorrect
  - Example: Newton's laws break down at subatomic scale
- We can assess the usefulness of representations only relative to the tasks we want to perform

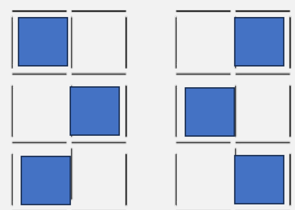
Sticks and Squares Problem

- We have 17 sticks arranged to form 6 squares
- Your task is to remove exactly 5 sticks so that we are left with exactly 3 squares and no orphan sticks



Sticks and Squares Problem

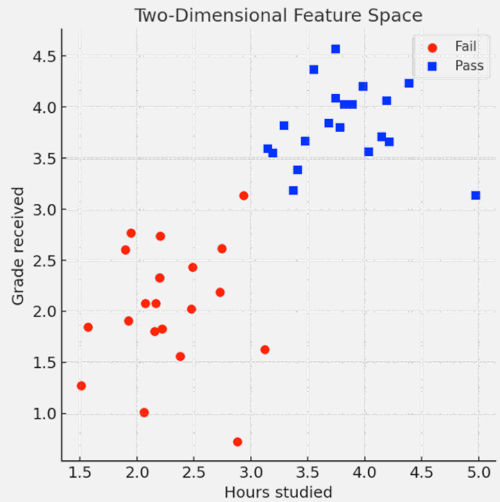
- We have 17 sticks arranged to form 6 squares
- Your task is to remove exactly 5 sticks so that we are left with exactly 3 squares and no orphan sticks
- Level of abstraction – sticks
  - $\binom{17}{5} = \frac{17!}{(17-5)!5!} = 6188$ configurations
- Level of abstraction – squares
  - $\binom{6}{3} = \frac{6!}{(6-3)!3!} = 20$ configurations



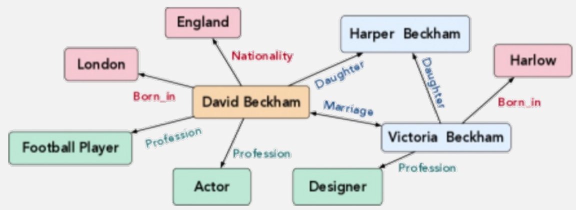
- Add some knowledge of geometry
  - Solution pops out

Feature spaces

- A feature space is constructed by assigning a problem-related attribute or feature or measurement to a distinct dimension of a multidimensional space.
- Points that are close together in feature space represent objects that have similar feature values
- Feature spaces are used by machine learning systems



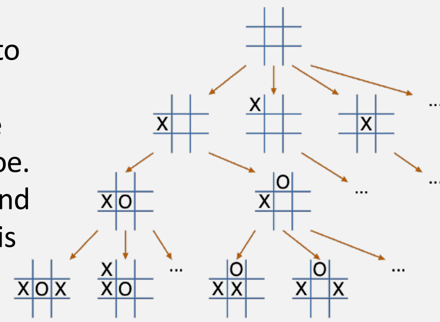
Relational networks



- Relational networks are graphs where the nodes represent individuals or other entities and links denote relationships between entities
- Such a representation can be used to answer queries about individuals and their relationships to other individuals or entities
  - Who is David Beckham married to?
  - Who is the daughter of David Beckham and Victoria Beckham?
  - Who is both an actor and a Football player? etc.

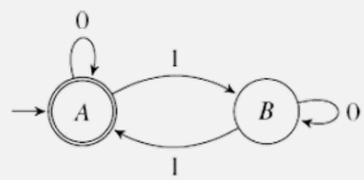
Game trees

- In a game tree, each node, representing a state, is connected to one or more successor states
- Figure shows the game tree for the familiar 2-person game of tic-tac-toe.
- Two players (X and O) take turns, and at each turn the player can place his or her mark (X or O) in an empty square.
- The goal is to traverse the tree from an initial state to a desired goal state.
- The player who gets three X's or three O's along a row, column, or diagonal wins.



State transition graphs

- This representation is a generalization of finite state automaton. The nodes denote states and labeled links denote conditions that the input must satisfy for the corresponding state transition to occur.
- The state diagram in Figure shows a ``even parity machine''



Logic

- The propositional and predicate logic, and their extensions, e.g., logics of knowledge and belief formalize of the process of inferring assertions about the world that are true from existing facts about the world.
- Given All men are mortal, that is, Socrates is a man, we can deduce using predicate logic that Socrates is mortal

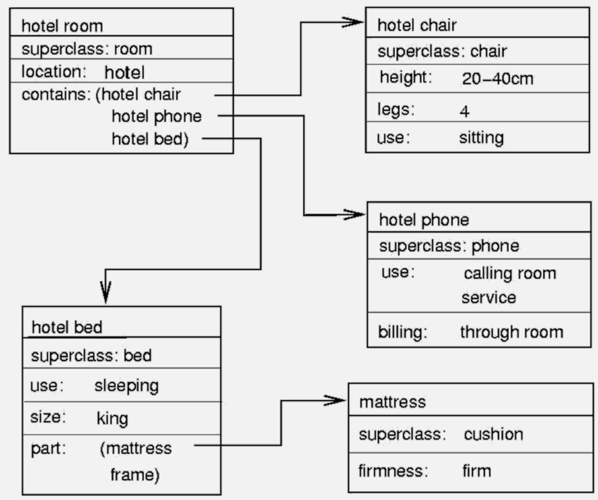
Procedural representation

- A procedural representation describes knowledge about the world in terms of a procedure that instructs an agent to perform a specific task.

Procedure Boil Water
(1) Obtain pot, and put water in it
(2) Put pot over the stove, and light the stove
(3) Turn off the stove when steam rises

Frame representation

- A frame provides a means of representing knowledge about prototypical objects, relationships, and events.
- The aspects of a given scenario are stored as entries in the slots of the frame.
- A partial frame representation of a hotel room is shown in Figure



Production(rule) systems

- Production systems use rules of the form IF condition THEN consequence
- Such rules form the basis of the so-called expert systems
- Applications range from configuring computer systems to medical diagnosis

Rules of the Road
- IF the stoplight is red AND there is no "No turn on Red" sign AND you have stopped, THEN a right turn is ok.
- IF the stoplight is green THEN a right turn is ok.
- IF right turn is ok THEN turn right.

Isomorphic representations

- Isomorphic representations enforce a direct structural relation between the representation and some of the properties of the world being represented.
- Road maps offer an example of a widely used isomorphic representation
- For any current location on the map, we might ask,
  - What is the nearest major town, and how far away is it?
  - What is the closest highway intersection where at least three roads meet?

KR is a set of ontological commitments

- What does an agent care about?
  - ✓Entities
    - coffee, liquid, tongue
  - ✓Properties
    - being hot, being able to burn
  - ✓Relationships
    - Coffee is a liquid
- KR involves abstraction, simplification
  - A representation is
    - a model of the world
    - like a cartoon
  - All representations are wrong, but some are useful

KR involves a set of epistemological commitments

- What can we know?
  - Boolean logic
    - Is a proposition True or False?
  - Probability theory
    - What is the probability that a given proposition true?
  - Decision theory
    - Which choice among a set of candidate choices is the most rational?
  - State-transition system
    - What is the next state for given current state and action?
  - Relational graphs
    - Relations between individuals
  - Feature spaces
    - Values of specific features for an individual

Knowledge representation is a theory of reasoning

- How can knowledge be encoded ?
  - Syntax
- What does the encoded knowledge mean?
  - Inferences that are sanctioned by the semantics
- What can we infer from what we know?
  - Inferences that can be performed by algorithms
- How can we manage inference?
  - What should we infer from among the things we can infer?

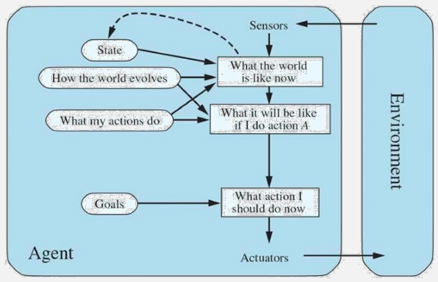
KR is a medium for efficient computation

- Reasoning = computation
- Anticipated by Leibnitz, Hilbert
  - Can all truths be reduced to calculation?
  - Is there an effective procedure for determining whether or not a conclusion is a logical consequence of a set of facts?
- KR involves tradeoffs between
  - Expressivity and tractability (decidability, efficiency) tradeoff
    - The more you can say (using a representational formalism), the less you can effectively do (within the formalism)
  - General purpose reasoning versus special-purpose, domain-specific inference
  - Declarative versus procedural knowledge

KR is a medium of expression and communication

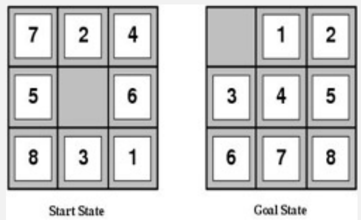
- If we assume shared
  - Ontological and epistemological commitments
  - KR formalism (syntax, semantics, reasoning)
- Then KR is a medium for
  - Expression
    - How general is it?
    - How precise is it?
    - Is it expressive enough?
  - Communication
    - Can we talk or think in the language?
    - Can we communicate the things we want to communicate?
    - What things are difficult to communicate?

Problem solving agents



- The agent is given a goal
- Agent is not told precisely how to achieve its goal
- Achieving a goal may require a long sequence of actions
- Agent needs to figure out how to achieve its goal
- Agent needs a model (representation) of the world and how its actions affect the world

Example: 8-puzzle



- States?
  - Position of each tile on the board
- Initial state?
  - Any state can be initial
- Actions?
  - {*Left*, *Right*, *Up*, *Down*}
- Goal test? Check whether goal configuration is reached

Problem Formulation

Simplifying assumptions

- Discrete, fully observable states
  - ‘in class’, ‘at home’
- Discrete actions
  - Mary executes action ‘Go home’ in state ‘in class’ to reach the ‘at home’ state
  - In this setup, we can’t speak of Mary being on her way home
- Passive environment
  - All state changes are due to the agent’s own actions
  - Mary can’t end up at home because her mom picked her up

Representation

A representation

- Is a surrogate inside an agent's 'brain' for entities that exist in the external world
- Derives its semantics through grounding (sensors, effectors)
- Embodies a set of ontological commitments – assumptions about the entities, properties, relationships, and actions that we care about

Properties of representations

- expressive power
- complexity, ..

Problem Solving Agents

Pizza Run: The Scenario

- Hungry Joe is relaxing at home on a Saturday afternoon
- He starts salivating about pizza for lunch
- He wants to be eating a pizza
- Time for a pizza run!

Pizza Run: The problem to be solved

- Current State: Hungry, sitting on the couch, at home
- Desired Goal State: Sitting at the table, eating pizza
- Solution: A plan or a sequence of actions that you can execute to go from the Current State to the desired Goal State

Pizza Run: The Plan

- Get off the couch, get dressed
- Grab wallet and car keys
- Leave apartment, walk to car
- Drive to nearest pizza place
- Order, pay, and pick up pizza
- Return home with pizza
- Sit down, open the box, enjoy!

How could Hungry Joe solve the problem?

- Given the start state and the goal state, and a set of actions at his disposal
  - Hungry Joe had to find a suitable sequence of actions
    - for a successful pizza run
    - that if correctly executed, would take him from the start state to the goal state

Generalizing the pizza run – state space search

- Many problems share the same structure
- The agent is given
  - a start state
  - a goal state
  - a set of actions, each with its own
    - Pre-conditions
    - Post-conditions
- Task: Find a sequence of actions that, if successfully executed, will take the agent from the start state to the desired goal state

State-space problem representation

A state space search problem is specified by

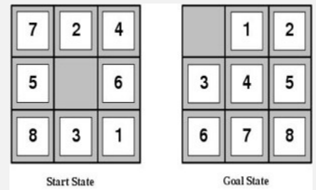
- *S is* the set of possible start states
- *O* is the set of actions
  - Partial functions that map a state into another
- *G* the set of goal states
  - *G* may be explicitly enumerated or implicitly specified using a goal predicate *goal* (*g*) = *True* iff *g* is a goal

Solution to a state space search problem is a sequence of actions leading from the start state $s \in S$ to a goal $g \in G$

State-space problem formulation

- Design a state representation that
  - Captures the relevant aspects of the world
  - Abstracts away unimportant details
- Formulate the goals
  - Specification of the solution
    - Explicit (enumeration of goal states)
    - Implicit (goal predicate)
- Formulate the actions
  - Preconditions
    - The conditions that need to hold before an action can be executed
  - Post-conditions
    - The conditions that are guaranteed to hold after an action is (successfully) executed

Example: 8-puzzle

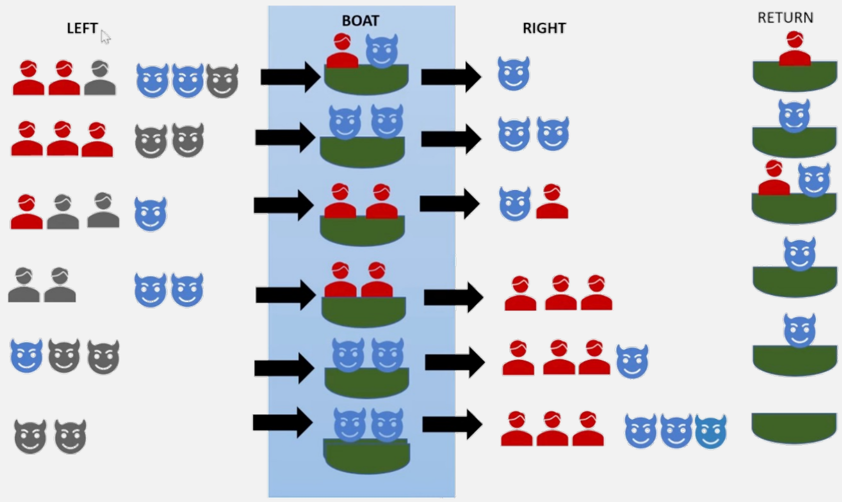


- States: Location of each tile on the board
- Initial state: Any state can be initial
- Actions:
  - *Left* – swap the blank tile with the tile to its left
  - *Right* – swap the blank tile with tile to its right
  - *Up* – swap the blank tile with tile above it
  - *Down* -- swap the blank tile with the tile below it
- Goal test: Check whether goal configuration is reached
- Solution: A sequence of actions

Example: Missionaries and Cannibals

- Initial state: 3 missionaries, 3 cannibals, and the boat on the left bank of the river
- Goal: all on the right bank
- Constraints:
  - The boat which can carry at most 2 people at a time
  - If missionaries are outnumbered by cannibals, the cannibals will eat the missionaries
- States: The positions of missionaries, cannibals, and the boat on either side of the river
- Actions: Movement of the boat with its occupants from one side of the river to the other, subject to the constraints
- Solution: A sequence of boat trips across the river complete with their passenger lists

Example: Missionaries and Cannibals

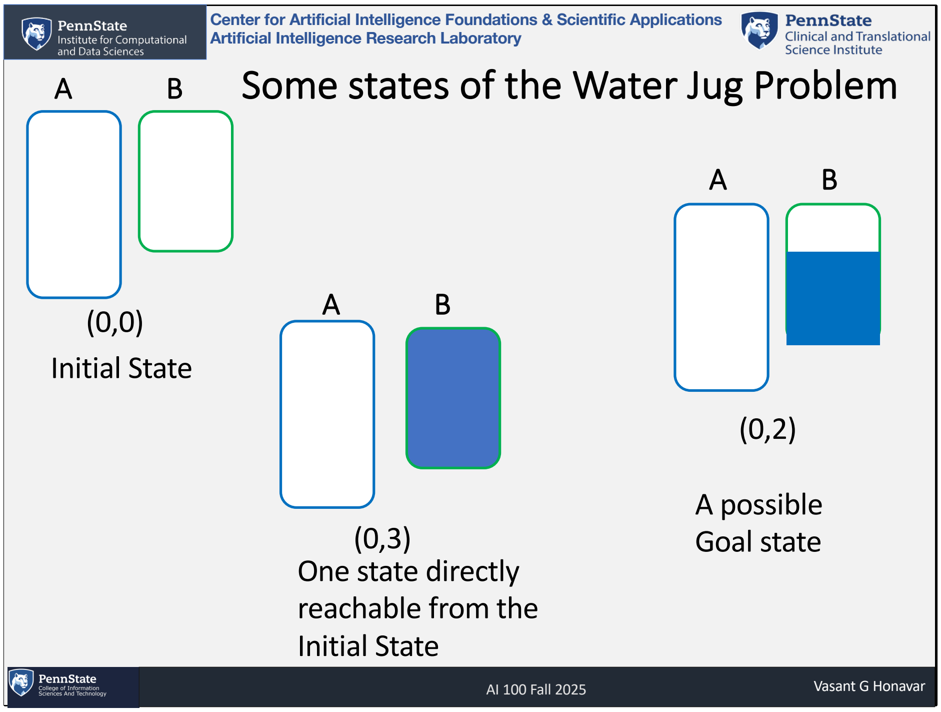


Water jug problem

- You have two jugs – A, B – with capacities of 4 liters and 3 liters
- Neither has measurement markings
- Actions:
  - Fill a jug completely
  - Empty a jug completely
  - Pour water from one jug to another until it is full or the first one is empty
- Goal: Measure exactly 2 liters of water

Water jug problem

- You have two jugs – A, B – with capacities of 4 liters and 3 liters
- Neither has measurement markings
- Actions:
  - Fill a jug completely
  - Empty a jug completely
  - Pour water from one jug to another until it is full or the first one is empty
- Goal: Measure exactly 2 liters of water
- Representation
  - States: (x, y), where x = water in Jug A, y = water in Jug B
  - Initial State: (0, 0) (both empty)
  - Goal State: (x, 2) (Jug B has 2 liters, A can have any amount)


PennState Institute for Computational and Data Sciences
Center for Artificial Intelligence Foundations & Scientific Applications
Artificial Intelligence Research Laboratory
PennState Clinical and Translational Science Institute
Some states of the Water Jug Problem
A
B
(0,0)
Initial State
A
B
(0,3)
One state directly reachable from the Initial State
A
B
(0,2)
A possible Goal state
PennState College of Information Sciences and Technology
AI 100 Fall 2025
Vasant G Honavar

A possible solution to the Water Jug problem

- You have two jugs – A, B – with capacities of 4 liters and 3 liters
- Actions:
  - Fill a jug completely
  - Empty a jug completely
  - Pour water from one jug to another until it is full or the first is empty
- Goal: Measure exactly 2 liters of water

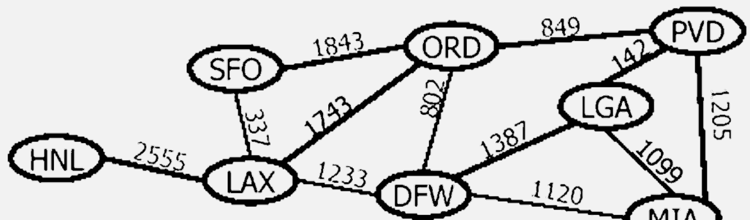
A possible solution

(0, 0) → Fill Jug B → (0, 3)→ Pour Jug B into Jug A → (3, 0)→ Fill Jug B → (3, 3) → Pour Jug B into Jug A → (4, 2) → Empty Jug A → (0, 2)

Example: Getting around in USA

- Working in SF
  - There is a graduation to attend in MIA
- Goal
  - To be in MIA
- Problem formulation
  - States: Various cities
  - Actions: Fly between cities connected by air
- Solution
  - Shortest route between SF and LA

Shortest path problem



Problem formulation in the observable, deterministic world

- A problem is defined by:
  - An initial state, e.g. SFO
  - Actions
    - SFO → ORD
    - SFO → LAX
    - …..
  - Goal test
    - Current state = MIA?
- Initial state + successor function defines a state space
- A solution is a sequence of actions from the initial to goal state

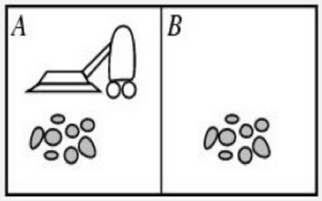
Basic State Space Search Problem

A state space search problem is specified by a 3-tuple (S, O, G) where

- S is a set of possible start states
- O is the set of actions (operators)
  - Partial functions that map a state into another
- G the set of goal states
  - G may be explicitly enumerated or implicitly specified using a goal test *goal* (g) = *True* iff g is a goal state

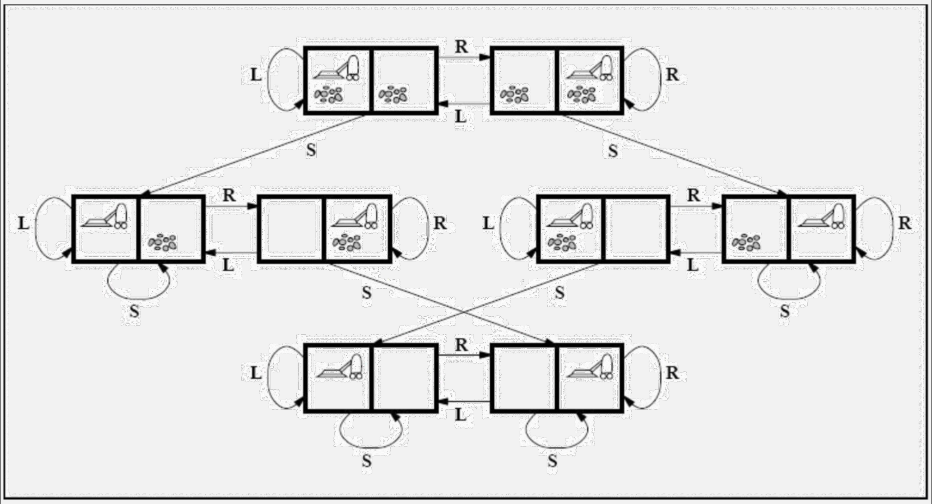
Solution to a state space search problem is a sequence of actions leading from the start state s to a goal $g \in G$

Example: vacuum world



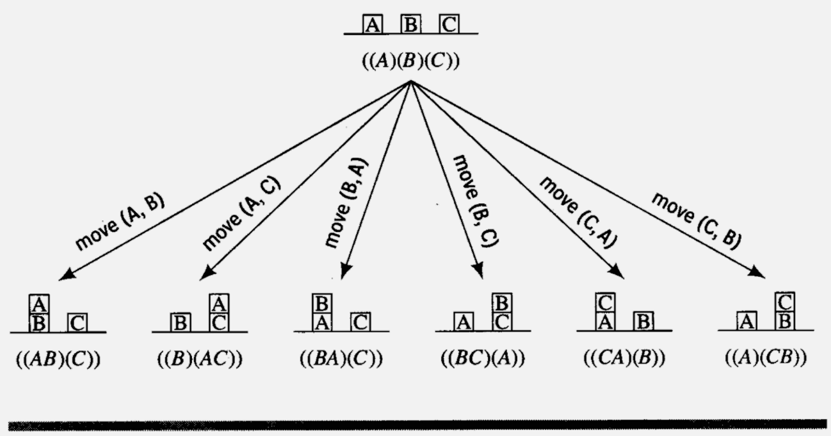
- States? two locations with or without dirt, with or without the vacuum cleaner: 2 x 2^2=8 states.
- Initial state? Any state can be initial
- Actions? {*Left*, *Right*, *Cleanup*}
- Goal test? Check whether both locations are clean.

Example: Complete State Space of the Vacuum World

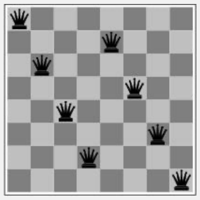


PennState Institute for Computational and Data Sciences
PennState Clinical and Translational Science Institute

Blocks World



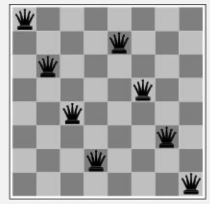
Example: 8-queens problem



- State: Any arrangement of Queens on the board
- Initial State: Empty board
- Actions: Placing a queen in an empty square
- Constraints:
  - Queens attack each other if they share
    - A row
    - A column
    - A diagonal
- Goal:
  - All 8 queens on the board with no queen attacking any other

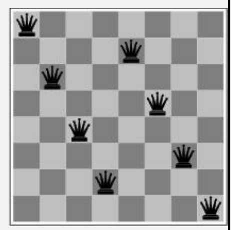
State space representation: 8-queens problem

Problem formulation 1



- State: Any arrangement of 0 to 8 queens on the board
- 64 squares, 8 queens
  - (64)(63)(62)(61)..(57) $\approx 3 \times 10^{14} \approx 1.2681 \times 2^{47}$ states!
- Can we do better?

State space representation: 8-queens problem

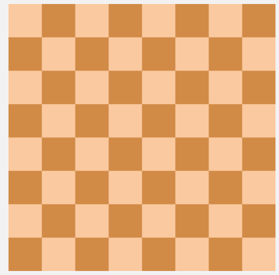


Problem Formulation 2

- State - any arrangement of 0 to 8 queens on the board
- 8 rows – need to specify the column in which a queen is placed in each row
  - Rows are variables which can take values in {1, 2, 3, 4, 5, 6, 7, 8}
  - $(8)(7)(6)(5)(4)(3)(2)(1) \approx 1.231 \times 2^{15}$ states!
  - Absorbed the `no two queens can share a row' constraint into the representation!
  - Much better than formulation 1 with 1.2681×2^{47} states!
  - Can we do better?

Example: 8-queens problem

Problem formulation 3



- State: Any arrangement of 0 to 8 queens on the board
- States: *n* (0≤ *n*≤ 8) queens on the board, one per column in the *n* leftmost columns with no queen attacking another
- Actions: Add a queen to the leftmost empty column so as not to attack the other queens
- Number of states = 2057

Representation matters!

Basic State Space Search Problem

A state space search problem is specified by

- *S is* the set of possible start states
- *O* is the set of actions (operators)
  - Partial functions that map a state into another
- *G* the set of goal states
  - *G* may be explicitly enumerated or implicitly specified using a goal predicate *goal* (*g*) = *True* iff *g* is a goal

Solution to a state space search problem is a sequence of actions leading from the start state *s* to a goal $g \in G$

Finding solution – State space search

Let L be a list of nodes yet to be expanded

1. Let *L* = list of partial paths to be extended
2. If *L* is *empty,* return *failure*
   else pick a partial path *p* from *L* (which node?)
3. If *p ends* in a goal node,
   a. return path from *s* to *p* and stop.
   b. Otherwise
      i. Delete p from *L*
      ii. Expand *p*: Add to *L* all of *p's* 1-step extensions (where?)
4. Return to 2.

Finding an optimal solution

- All actions may not be equally expensive
- Suppose we have a cost for each step
- $c(q,o,r)$ = cost of applying operator o in state q to reach state r
- Path cost is typically assumed to be the sum of costs of operator applications along the path
- An optimal solution is one with the lowest cost path from the specified start state s to a goal $g \in G$

Basic Search strategies

A search strategy specifies a particular order of node expansion
Search strategies are evaluated in terms of:

- Completeness:
  - Does it always find a solution if one exists?
- Admissibility
  - Does it always find an optimal solution?
- Time Complexity
  - Number of partial paths explored
  - Memory needed to store *L* during search
- Optimality
  - Optimal in its use of space, time, or both

Uninformed (blind) search strategies

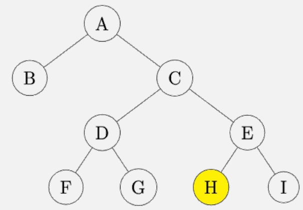
- Use only information available in problem definition
- Blind search strategies:
  - Breadth-first search
  - Depth-first search
  - Iterative deepening search
  - Bidirectional search

Some basic search algorithms

- Note 1
  - To illustrate the algorithm, we show the whole state space
  - But the agent does not see the whole state space
  - It starts in the initial state and must discover the rest of the state space through exploration
  - Different algorithms differ in terms how – in which order – they explore the states
- Note 2
  - We use "expand a node" and "extend a (partial) path" interchangeably

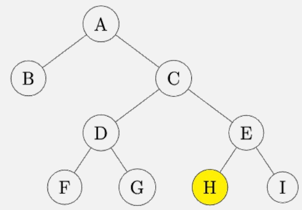
Breadth first search

- Breadth-first search starts with the start state as the root of the search tree
- Expand the start state, then expand its successors, then their successors …
  - at each step testing whether the corresponding state is a goal state,
  - until one of the nodes generated passes the goal test or we reach a dead end.
  - At any step if a node tested is not a goal node and has no successors, it is dropped from the list of partial paths.



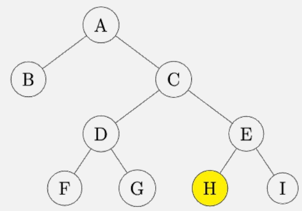
Breadth first search

A
AB, AC
ACD, ACE
ACDF, ACDG, ACEH, ACEI
ACDG ACEH ACEI
ACEH ACEI



If the maximum branching factor is finite, BFS is guaranteed to find a solution if one exists
Memory – exponential in the depth of the tree

Depth first search



A
AB AC
AC
ACD ACE
ACDF, ACDG, ACE
ACDG ACE
ACE
ACEH ACEI

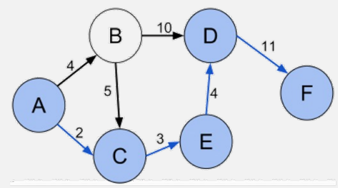
- DFS is guaranteed to find a solution if one exists only if the search space is finite and branching factor is finite
- Can fail to terminate if the search space is infinite
- Memory – linear in the depth of the tree

Iterative deepening search

- Do depth first searches with depth limited to 0, 1, 2, …
- If there is a solution at depth *d*, it will be found with DFS with depth cutoff *d*
- Are we increasing work?
  - Yes, but by a small factor because most of the work is done at depth *k* if *k* is the cutoff
  - IDS is guaranteed to find a solution if it exists (assuming branching factor is finite)

Finding optimal solutions: Branch and Bound Search

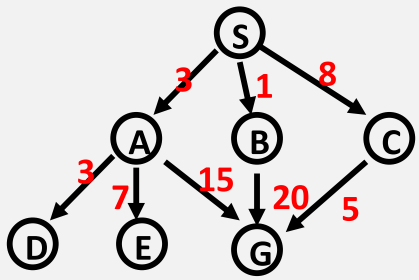
1. A (0)
2. AC (2), AB (4)
3. AB(4), ACE(5)
4. ACE(5), ABC(9), ABD(14)
5. ACED(9), ABC(9), ABD(14)
6. ABC(9), ABD(14), ACEDF(20)
7. ABCE(12), ABD(14), ACEDF(20)
8. ABD(14), ABCED(16), ACEDF(20)
9. ABCED(16), ACEDF(20), ABDF(25)
10. ACEDF(20), ABDF(25), ABCEDF(27)



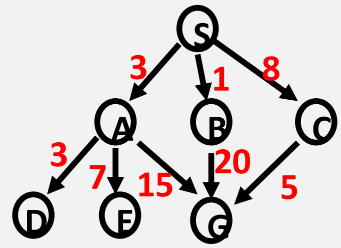
- Now the first path on the list takes us from A to F.
- Because the list is sorted in increasing order of cost, the cost of ACEDF is $\leq$ the costs of all other paths on the list
- Hence, it must be the optimal (cheapest) path from A to F

Exercise:

Suppose G is the goal node
Use DFS, BFS and Branch and Bound Search
Which of them is guaranteed to find the optimal solution?



Depth-First Search

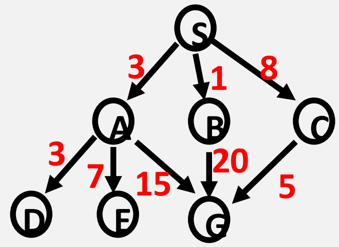


Expanded node	Paths list
	{ S^0 }
S^0	{ SA^3 SB^1 SC^8 }
A^3	{ SAD^6 SAE^{10} SAG^{18} SB^1 SC^8 }
D^6	{ SAE^{10} SAG^{18} SB^1 SCG^{13} }
E^{10}	{ SAG^{18} SB^1 SC^8 }
G^{18}	{ SB^1 SC^8 }

Solution path found is S A G, cost 18

Number of nodes expanded (including goal node) = 5

Breadth-First Search

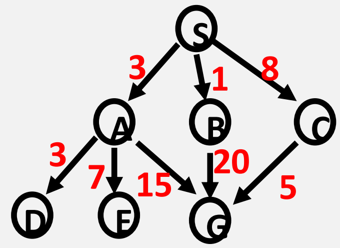


Expanded node	Partial paths list
	{ S^0 }
S^0	{ SA^3 SB^1 SC^8 }
A^3	{ SAB^1 SAC^8 SAD^6 SAE^{10} SAG^{18} }
B^1	{ SAC^8 SAD^6 SAE^{10} SAG^{18} SBG^{21} SCG^{13} }
C^8	{ SAD^6 SAE^{10} SAG^{18} SBG^{21} SCG^{13} }
D^6	{ SAE^{10} SAG^{18} SBG^{21} SCG^{13} }
E^{10}	{ SAG^{18} SBG^{21} SCG^{13} }
G^{18}	{ SAG^{21} SCG^{13} }

Solution path found is S A G , cost 18

Number of nodes expanded (including goal node) = 7

Branch and Bound Search



Expanded node	Paths list
	{ S^0 }
S^0	{SB^1 SA^3 SC^8 }
B^1	{SA^3 SC^8 SBG^{21}}
A^3	{ SAD^6 SC^8 SAE^{10} SAG^{18} SBG^{21}}
D^6	{SC^8 SAE^{10} SAG^{18} SBG^{21}}
C^8	{SAE^{10} SCG^{13} SAG^{18} SBG^{21} }
E^8	{ SCG^{13} SAG^{18} SBG^{21} }

Solution path found is S A G, cost 13

Number of nodes expanded (including goal node) = 7

Bidirectional search