PennState Institute for Computational and Data Sciences

Center for Artificial Intelligence Foundations & Scientific Applications
Artificial Intelligence Research Laboratory

PennState Clinical and Translational Science Institute

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

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 formalisms

- Feature spaces
- Relational graphs
- Game trees
- State transition systems
- Logic
- Rules
- Frames
- Procedures
- Probabilities
- Isomorphic representations
- .....

KR formalisms

- Logical
  - e.g., Propositional Logic, First order logic, Description logic
- Probabilistic
  - e.g., Bayesian networks
- Grammatical
  - e.g., Context free grammars
- Structured
  - e.g., frames – as in object-oriented programming
- Decision theoretic
- …

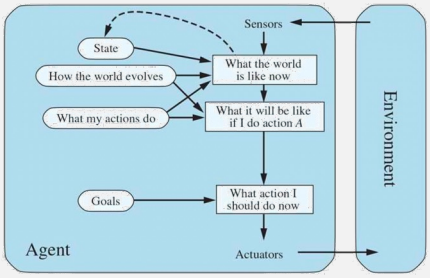
KR is a medium for efficient computation

- 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

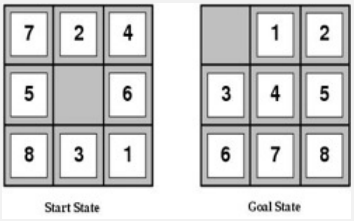
- If we assume shared
  - Ontological and epistemological commitments
  - KR formalism (syntax, semantics, reasoning)
- Then KR is a medium for
  - Expression
    - How general is it?
    - How precise is it?
    - Is it expressive enough?
  - Communication
    - Can we talk or think in the language?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

- Maps each (physical) state of the external environment into the corresponding abstract state via sensors
- Maps each (physical) action on an environmental state into an abstract action on the corresponding abstract state
- Maps effects of an abstract action on an abstract state into a corresponding effect on the corresponding physical state via effectors

The mapping from

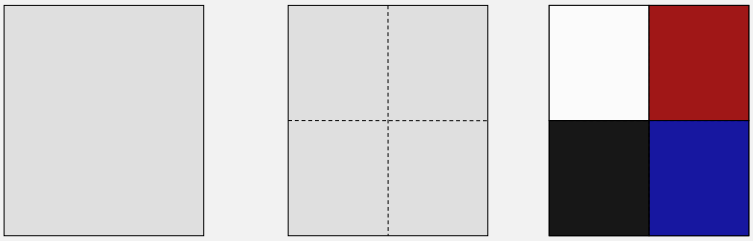
- environmental states and abstract states is many to one
- abstract state to an environmental state is one to many

Representation

The mapping from

- environmental states and abstract states is many to one
- abstract state to an environmental state is one to many

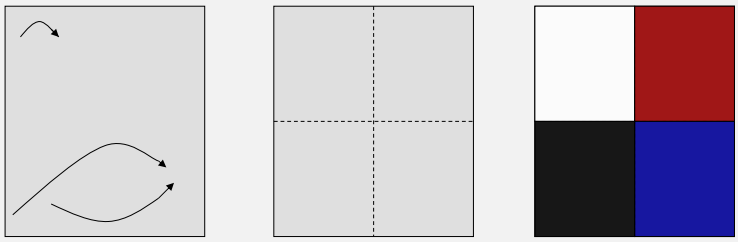
A representation induces a partition over environmental states



- 4 abstract states, 12 abstract actions (lateral, vertical, or diagonal moves)
- 4 abstract states, 8 abstract actions (lateral, vertical but not diagonal moves

Representation

- Effects of abstract actions in the abstract state space is fully deterministic and predictable.. But..
- The corresponding effects of the physical actions on the world are predictable to the extent allowed by
  - the resolution of the representation and
  - the fidelity of sensors and effectors



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 Formulation

- Formulate the goals
  - Explicit specification
  - Implicit specification (goal predicate)
- Formulate the actions
  - Preconditions (before)
  - Post-conditions (after)
- Design a representation that
  - Captures the relevant aspects of the world
  - Abstracts away unimportant details

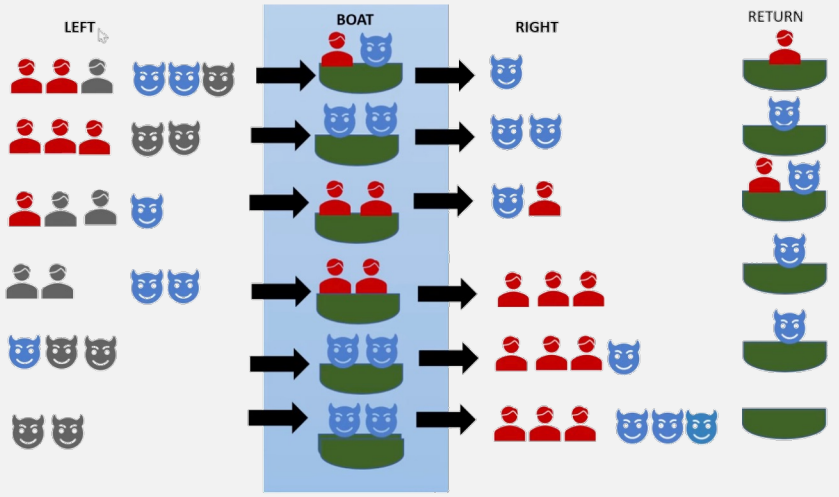
Problem Formulation

- Formulate the goals
  - Explicit specification (enumeration of goal states)
  - Implicit specification (goal predicate)
- Formulate the actions
  - Preconditions (before)
  - Post-conditions (after)
- Design a representation that
  - Captures the relevant aspects of the world
  - Abstracts away unimportant details

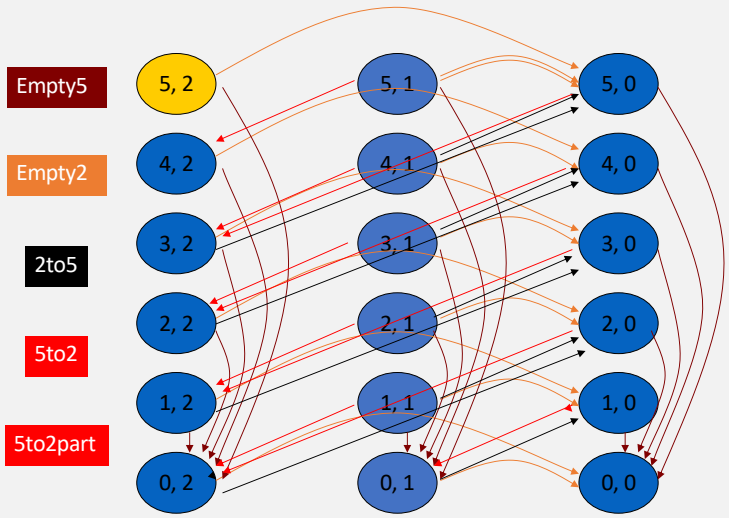
Example: Missionaries and Cannibals

- 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
- Solution: A sequence of boat trips across the river complete with their passenger lists

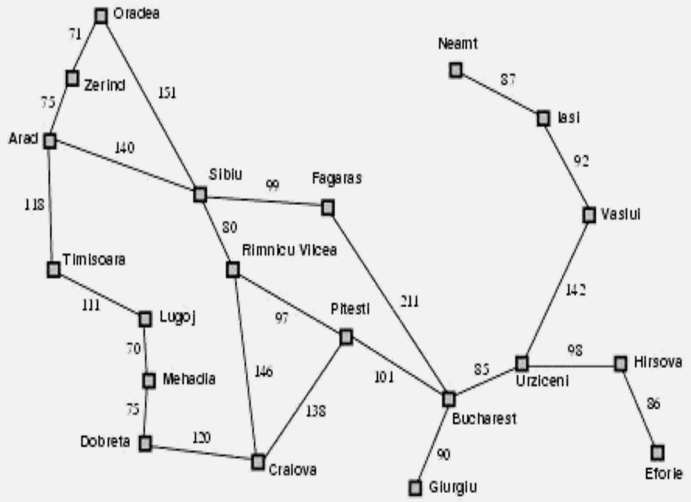
Example: Missionaries and Cannibals



Water jug state space



Example: Getting around in Romania



Example: Getting around in Romania

- On holiday in Romania; currently in Arad
  - Flight leaves tomorrow from Bucharest
- Goal
  - To be in Bucharest
- Problem formulation
  - States: various cities
  - Actions: drive between cities
- Solution
  - Sequence of cities; e.g. Arad, Sibiu, Fagaras, Bucharest

Problem formulation in the observable, deterministic world

- A problem is defined by:
  - An initial state, e.g. *Arad*
  - Actions
    - Arad → Zerind
    - Arad → Sibiu
    - Arad → Timisoara
  - Goal test
    - Current state = Bucharest?
- 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

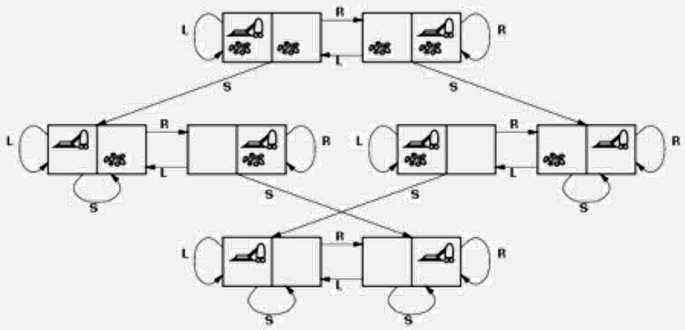
A state space search problem is specified by a 3-tuple

(*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

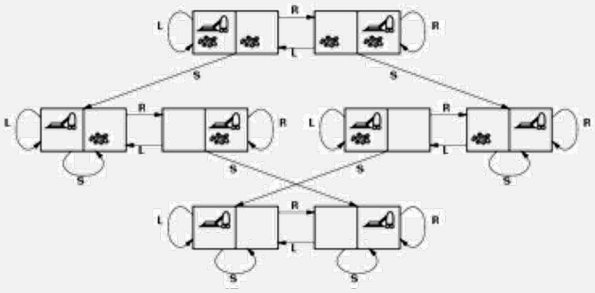
Solution to a state space search problem is a sequence of actions leading from the start state *s* to a goal $g \in G$

Example: vacuum world



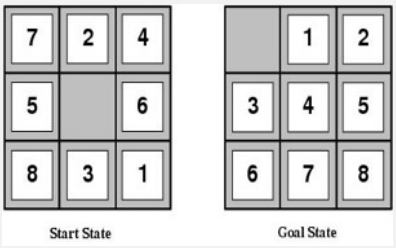
- States?
- Initial state?
- Actions?
- Goal test?
- Path cost?

Example: vacuum world



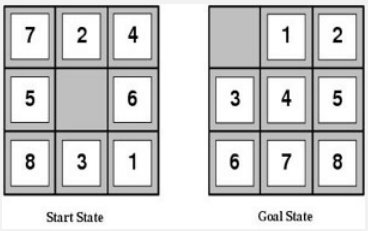
- States? two locations with or without dirt, with or without the vacuum cleaner: 2 x 2^2=8 states.
- Initial state? Any state can be initial
- Actions? {*Left*, *Right*, *Cleanup*}
- Goal test? Check whether both locations are clean.
- Path cost? Number of actions to reach goal

Example: 8-puzzle



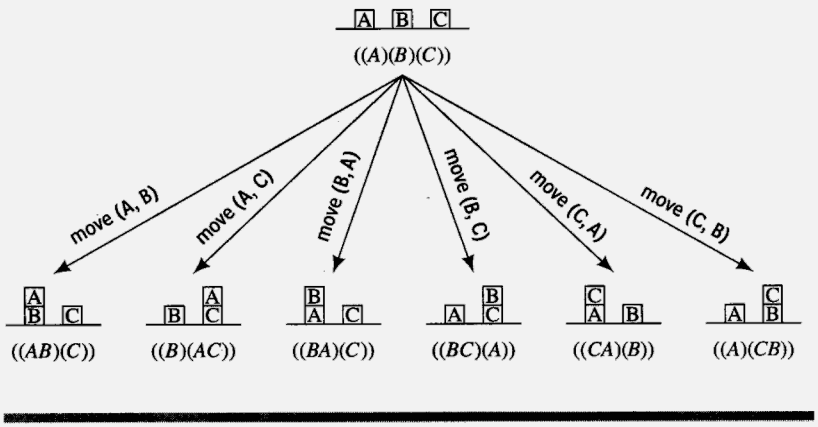
- States?
- Initial state?
- Actions?
- Goal test?

Example: 8-puzzle

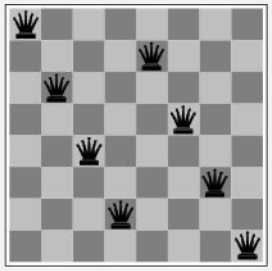


- States? Integer location of each tile
- Initial state? Any state can be initial
- Actions? {*Left*, *Right*, *Up*, *Down*}
- Goal test? Check whether goal configuration is reached
- Path cost? Number of actions to reach goal

Blocks World



Example: 8-queens problem



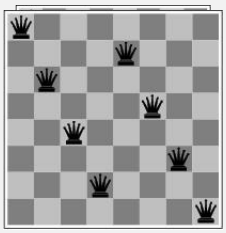
Constraints:

No two queens can share

- A row
- A column
- A diagonal

- States?
- Initial state?
- Actions?
- Goal test?

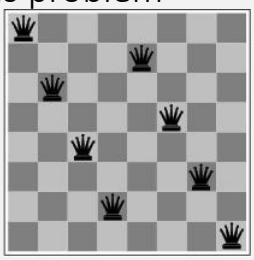
Example: 8-queens problem



Problem formulation

- States?
  - Any arrangement of 0 to 8 queens on the board
- Initial state?
  - Empty board (no queens)
- Actions?
  - Add a queen in an empty square
- Goal test?
  - 8 queens on board and none under attack

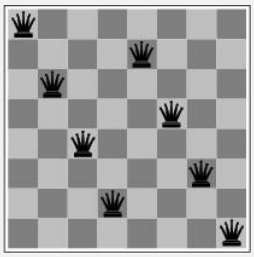
State space representation: 8-queens problem



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

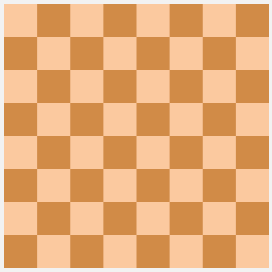


Problem Formulation 2

- 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
  - (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 ($0 \leq n \leq 8$) 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!

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.

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 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

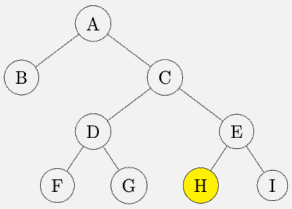
Search strategies are evaluated in terms of:

- 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

Some basic search algorithms

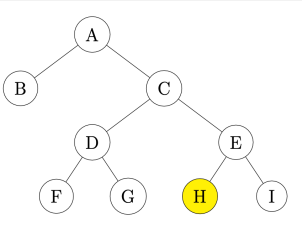
Breadth first search

- 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

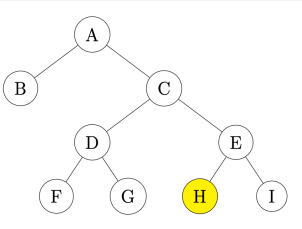
A
AB, AC
ACD, ACE
ACDF, ACDG, ACEH, ACEI
ACDG ACEH ACEI
ACEH ACEI



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



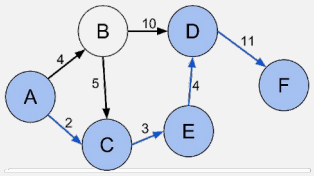
DFS is guaranteed to find a solution if one exists only if the search space is finite and branching factor is finite
Can fail to terminate if the search space is infinite
Memory – linear in the depth of the tree

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)

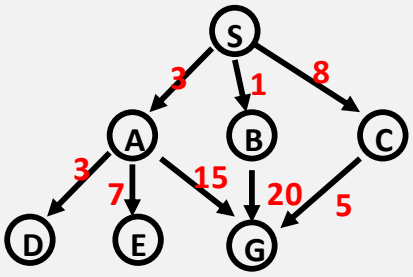
Finding optimal solution

(AC, 2), (AB, 4)
(AB, 4), (ACE, 5)
(ACE, 5), (ABC, 9), (ABD, 14)
(ACED, D), (ABC, 9), (ABD, 14)
(ABC, 9), (ABD, 14), (ACEDF, 20)
(ABCE, 12), (ABD, 14), (ACEDF, 20)
(ABD, 14), (ABCED, 16), (ACEDF, 20)
(ABCED, 16), (ACEDF 20), (ABDF, 25)
(ACEDF, 20), (ABDF, 25), ABCEDF, 27)



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

Bidirectional search