Introduction to Artificial Intelligence
COMP 3501 / COMP 4704-4
Lecture 2: Search

Prof. Nathan Sturtevant

Class Overview

Problem Solving Agents

- Requires a goal
- Requires actions
  - What actions?
- Requires state representation
  - How should state be represented?

Problem Solving Agents: Assumptions

- Assume world is:
  - Observable
  - Discrete
  - Known
  - Deterministic

Problem Solving Agents: Approach

- General approach is called “search”
- Input: *environment*, start state, goal state
  - Env.: states, actions, transitions, costs, goal test
- Output: sequence of actions
- Actions are executed after planning
  - Percepts are ignored when executing plan

Sample Domains

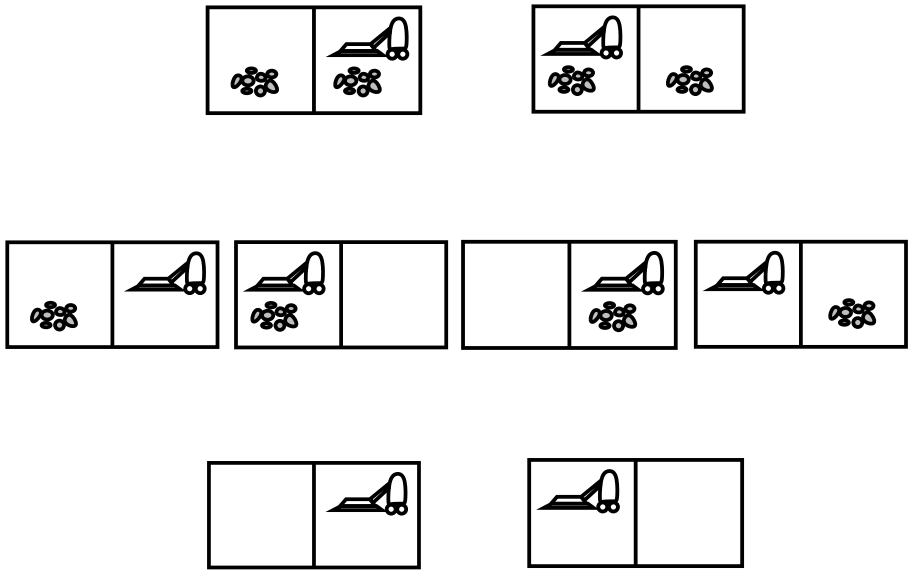
- Vacuum world
- Sliding-tile puzzle
- 8-queen puzzle
- Path planning

Vacuum world

- States:
- Initial state:
- Actions:
- Transitions:
- Goal test:
- Action Cost:

Vacuum world

- States: All combinations of agent & dirt locations [8]
- Initial state: Any state
- Actions: Left / Right / Suck
- Transitions: Left / Right put you in Left / Right cell
  - Suck removes dirt
- Goal test: No dirt
- Action Cost: 1 for all actions



Sliding Tile Puzzle

- States:
- Initial state:
- Actions:
- Transitions:
- Goal test:
- Action Cost:

Path planning

- States:
- Initial state:
- Actions:
- Transitions:
- Goal test:
- Action Cost:

Path planning variations

- Traveling sales problem
- Rectangle packing
- Robot navigation
- Multi-agent planning

Search Terminology

- Search tree: implicit/explicit set of searched states
- Node: single state in tree
  - Multiple nodes may represent the same state
- Expansion: generating the neighbors of a state
- Children: new neighbors of a state
- Parent: state from which neighbors were generated

General Best-First Search

- Open list: set of states considered next for expansion
  - Also called “search frontier”
  - States are ordered by some priority of “best”
- Closed list: set of states which have been expanded
  - Not all algorithms maintain a closed list

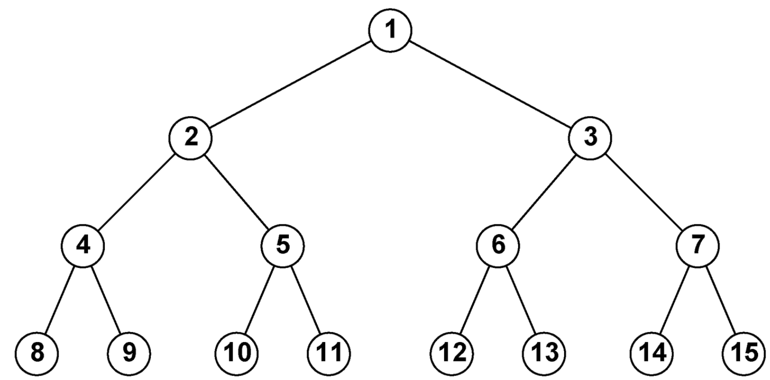
Algorithm Performance Measures

- Completeness:
  - Will we always find a solution when one exists?
- Optimality:
  - Will we find the shortest possible solution?
- Time complexity:
  - How long will it take to find a solution?
- Space complexity:
  - How much storage is required?

Uninformed search strategies

Breadth-first search

- Special case of best-first search
  - Best is by minimum depth
- Can be implemented by a FIFO queue



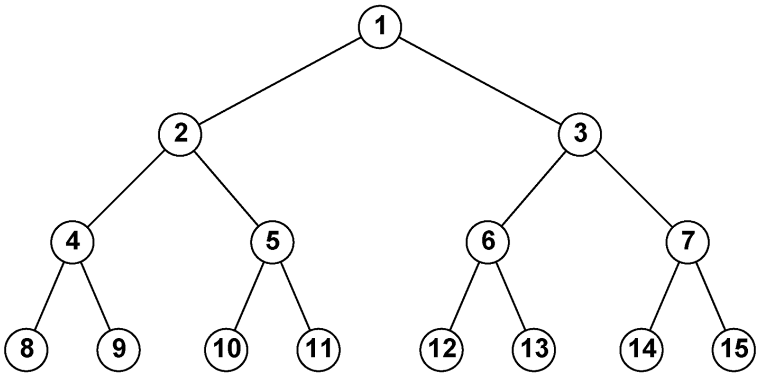
Breadth-first search

- Complete?
- Optimal?
- Time complexity?
- Space complexity?

- Implications of space complexity?

Depth-first search

- Special case of best-first search
  - Best is by *maximum* depth
- Can be implemented by a LIFO queue



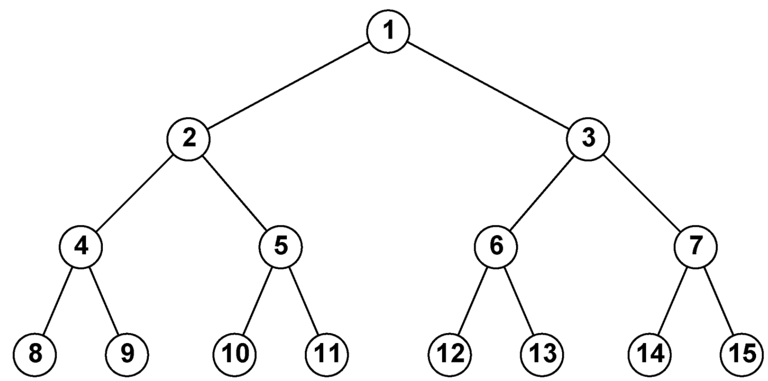
Depth-first search

- Complete?
- Optimal?
- Time complexity?
- Space complexity?

- Implications for infinite graphs?

Depth-first iterative deepening

- Iterated depth-first search
  - Iteratively perform depth-first search
  - Each iteration has a depth bound
  - Gradually increase depth bound until a solution is found

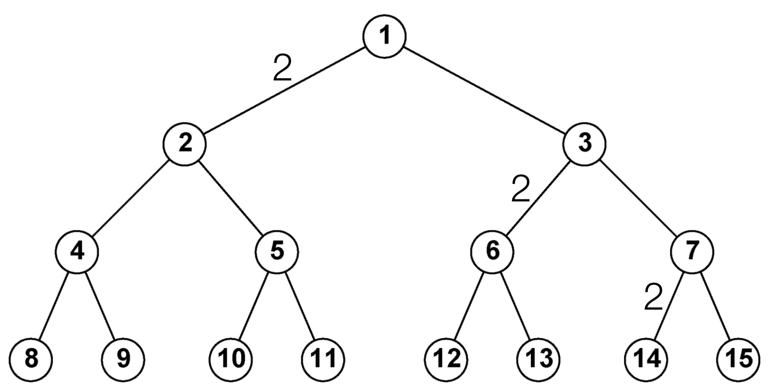


Depth-first search

- Complete?
- Optimal?
- Time complexity?
- Space complexity?

Uniform-cost (Dijkstra) search

- Special case of best-first search
  - Best is by minimum *cost*
- Priority queue needed to sort nodes by cost
  - *g*-cost is the cost from the start to current state



Uniform-cost search

- Complete?
- Optimal?
- Time complexity?
- Space complexity?

Uninformed vs. informed search

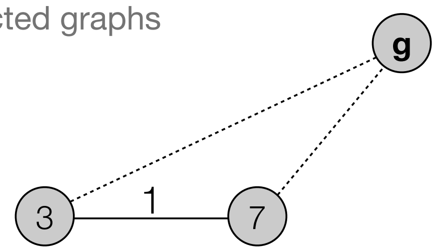
- Previous approaches were goal agnostic
  - Given the same start state the search is identical
- Incorporate information about the goal into the search

Heuristic function

- A heuristic estimates the cost to the goal from a state
  - h(s) or h(s, g)
- We are interested in *admissible* heuristics
  - Where h*(s) is a perfect heuristic
  - For an admissible heuristic h(s) ≤ h*(s) for all s.

Heuristic function

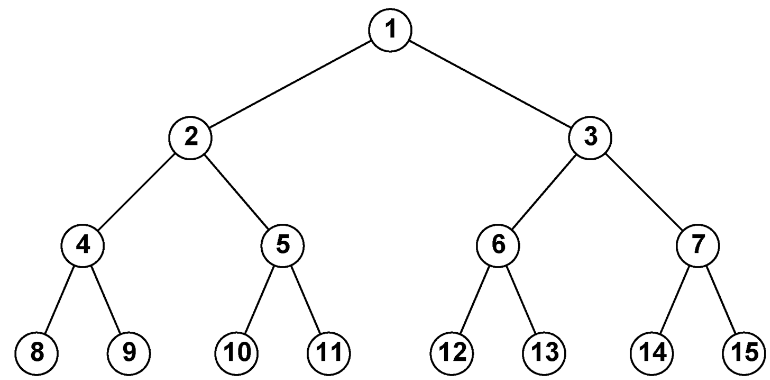
- Sometimes assume a heuristic is consistent
  - Obeys the triangle inequality
  - |h(a) - h(b)| ≤ c(a, b)
    - For undirected graphs



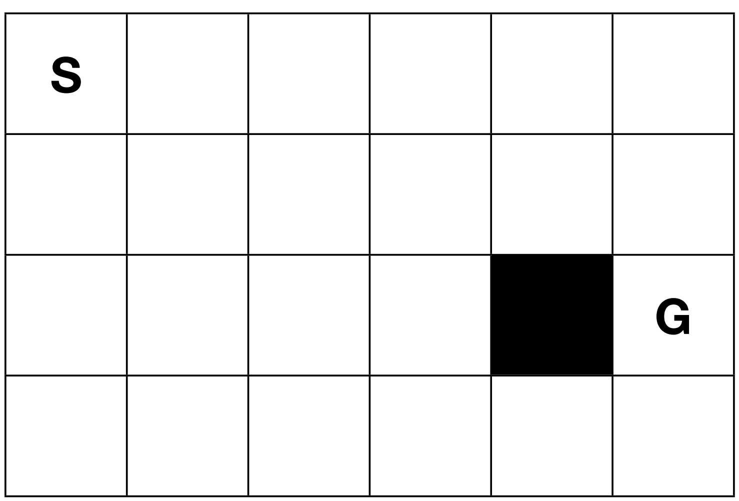
Informed search strategies

Greedy best-first search (Pure heuristic search)

- Special case of best-first search
  - Best is by minimum *heuristic value*
- Priority queue needed to sort nodes by cost
  - *h*-cost is the cost from the start to current state



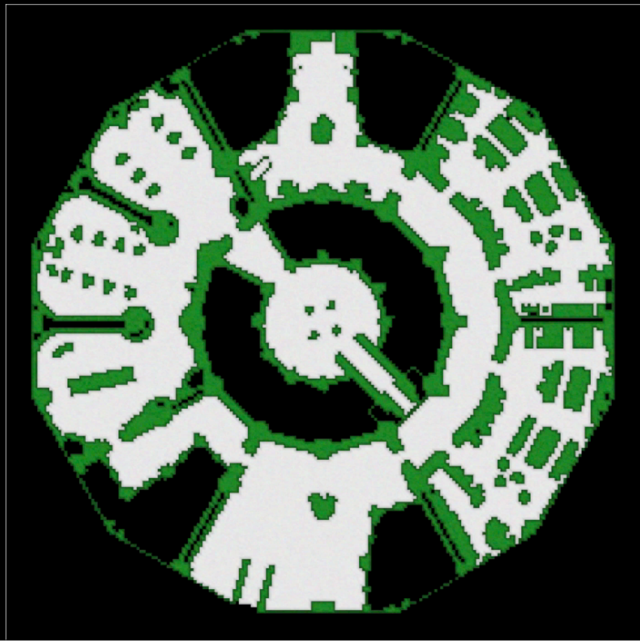
Assume heuristic is distance from leaves



Greedy best-first search

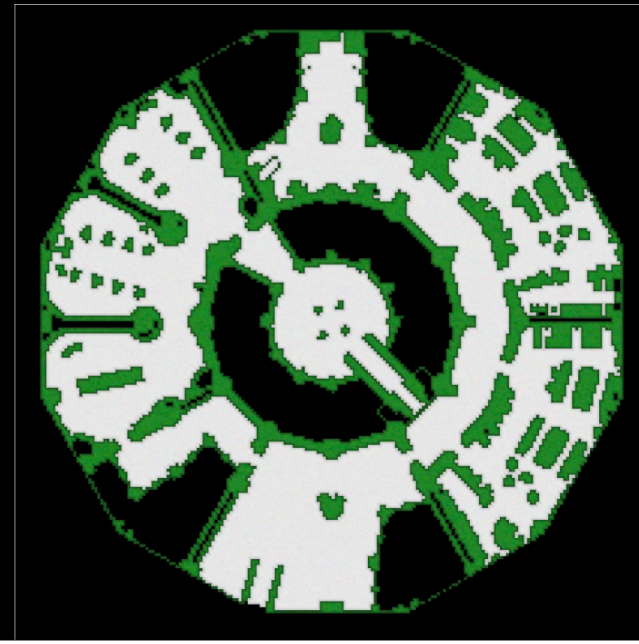
- Complete?
- Optimal?
- Time complexity?
- Space complexity?

Uniform Cost Search



37

Greedy Best-First Search



38

Homework: Problem 3.14

39