Inteligencia Computacional
Búsqueda

Gregorio Toscano Pulido

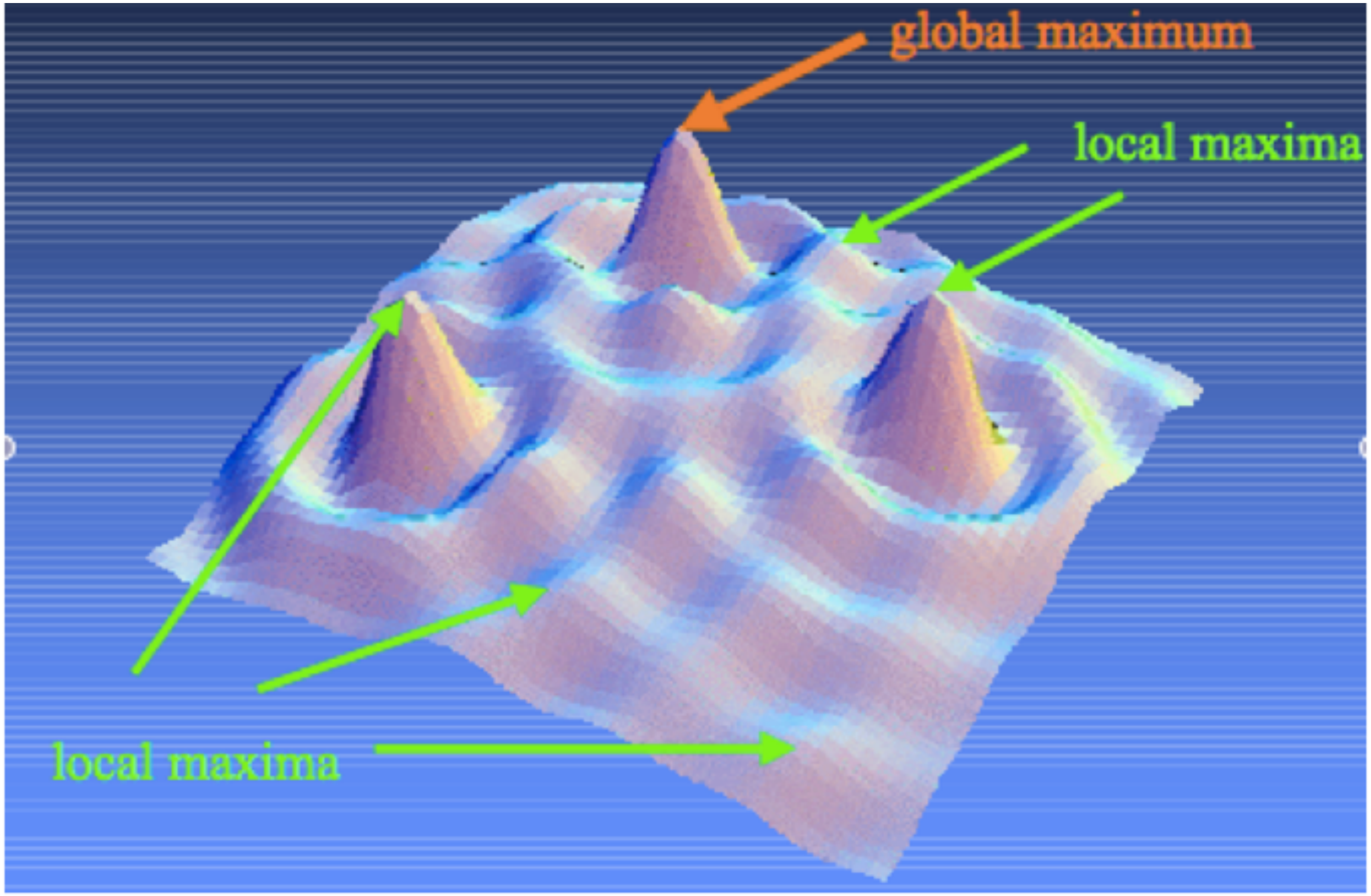
There are two types of problems:

- Calculation problems
  - sin(20), 3*4+5
- Search problems:
  - Eight queens problem
  - Rubik's cube
  - The towers of Hanoi
  - Traveling salesman problem

Definitions

- Problem solving is a process of generating solutions from the observed data.
- Problem space: a complete set of possible states, generated by exploring all possible steps, or moves, which may or may not lead from a given state or start state to a goal state.
- Solution: Refers to a combination of operations and objects that achieve the goals.
- Search: It refers to the search for a solution in a problem space.

Search space



What should we know for a preliminary

- Analysis of a search problem?
- What are the givens? Do we have all of them?
- Are the givens specific to a particular situation?
- Can we generalize?
- Is there a notation that represents the givens and other states succinctly?

What should we know for a preliminary analysis of a search problem (cont)

- What is the goal?
- Is there a single goal, or are there several?
- If there is a single goal, can it be split into pieces?
- If there are several goals or subgoals, are they independent or are they connected?
- Are there any constraints on developing a solution?

Components of a State Space Graph:

- Start: description with which to label the start node
- Operators: functions that transform from one state to another, within the constraints of the search problem
- Goal condition: state description(s) that correspond(s) to goal state(s)

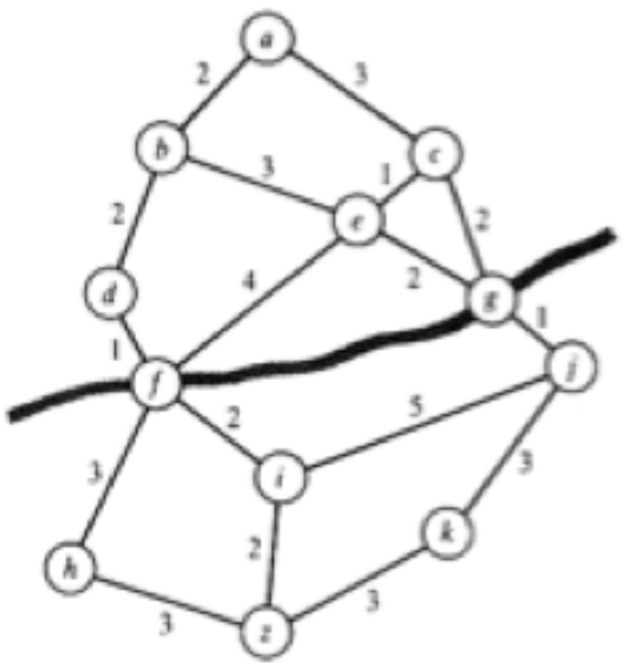
Basic problem solving strategies:

1. Basic search techniques
2. Problem decomposition and AND/OR graphs graphs
3. Searching with problem-specific knowledge

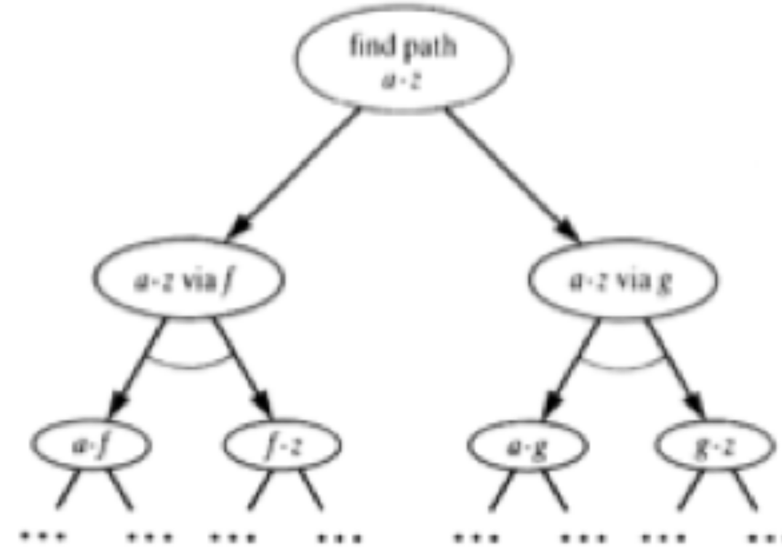
Basic search techniques:

- Depth-first vs. breadth-first search
- Greedy search
- Gradient descent or ascent
- Stochastic search
  - Simulated annealing
  - Evolutionary search

Problem decomposition and AND/OR graphs graphs



[Bratko,2001]



[Bratko,2001]

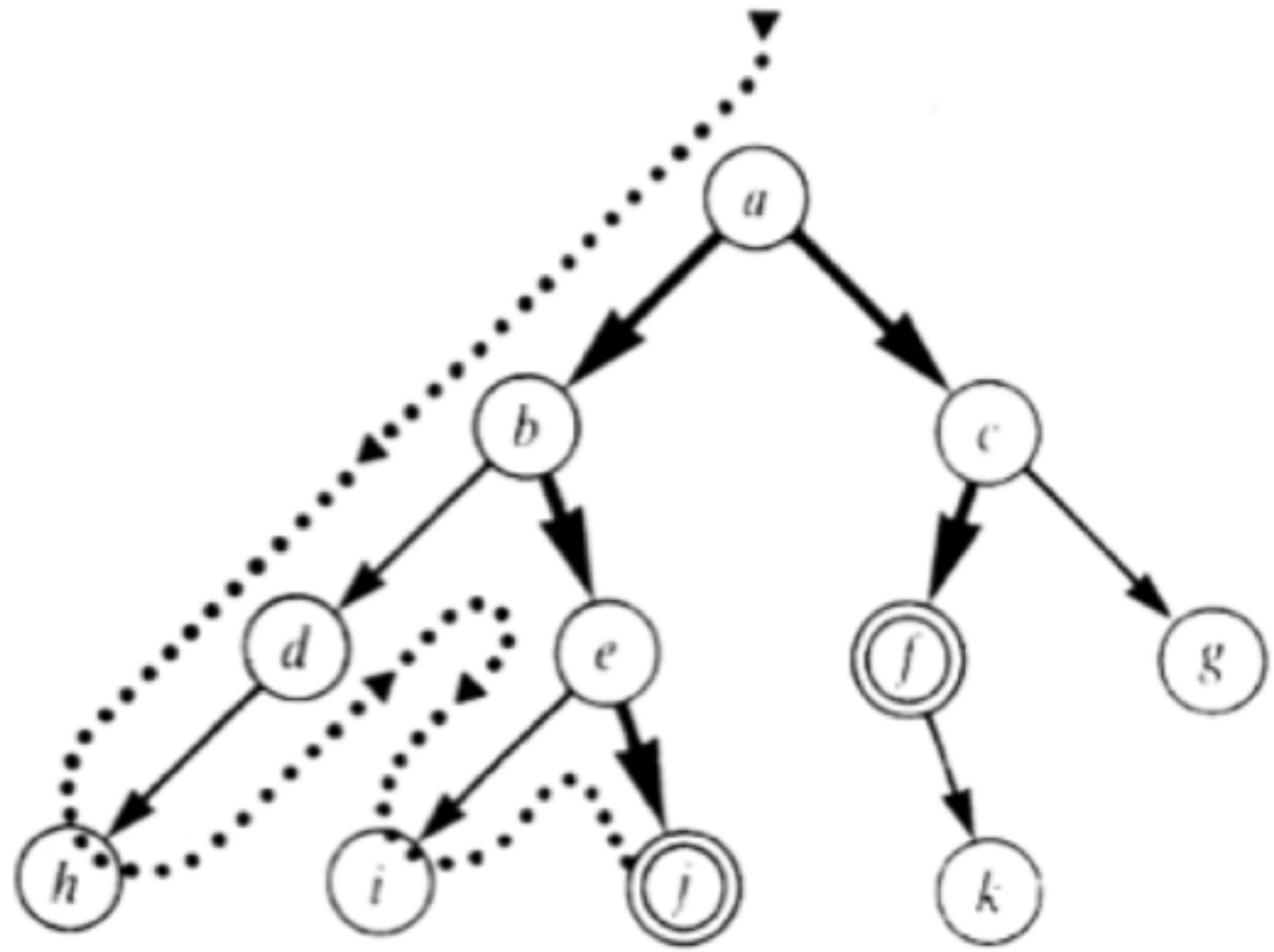
Searching with problem-specific knowledge

- Chess game
- Checkers game
- Chinese checkers game
- etc.

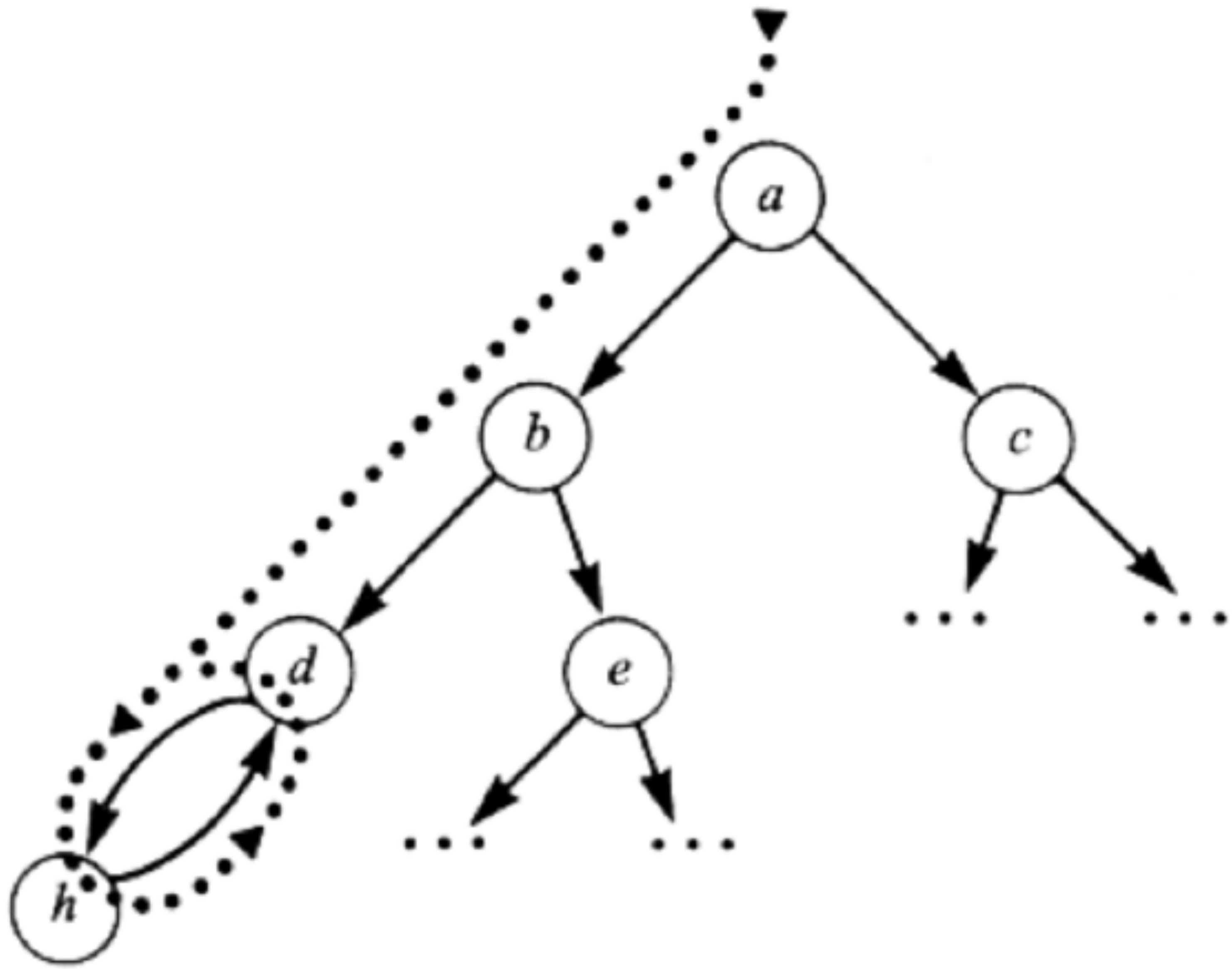
Which classification belong to?

- Hanoi towers,
- 8 puzzle,
- 8 queen problem,
- Rubik's cube,
- Impossible cube.

Basic search techniques: Depth-first search



Depth-first search: cycle detection



Depth-first search: advantages and drawbacks

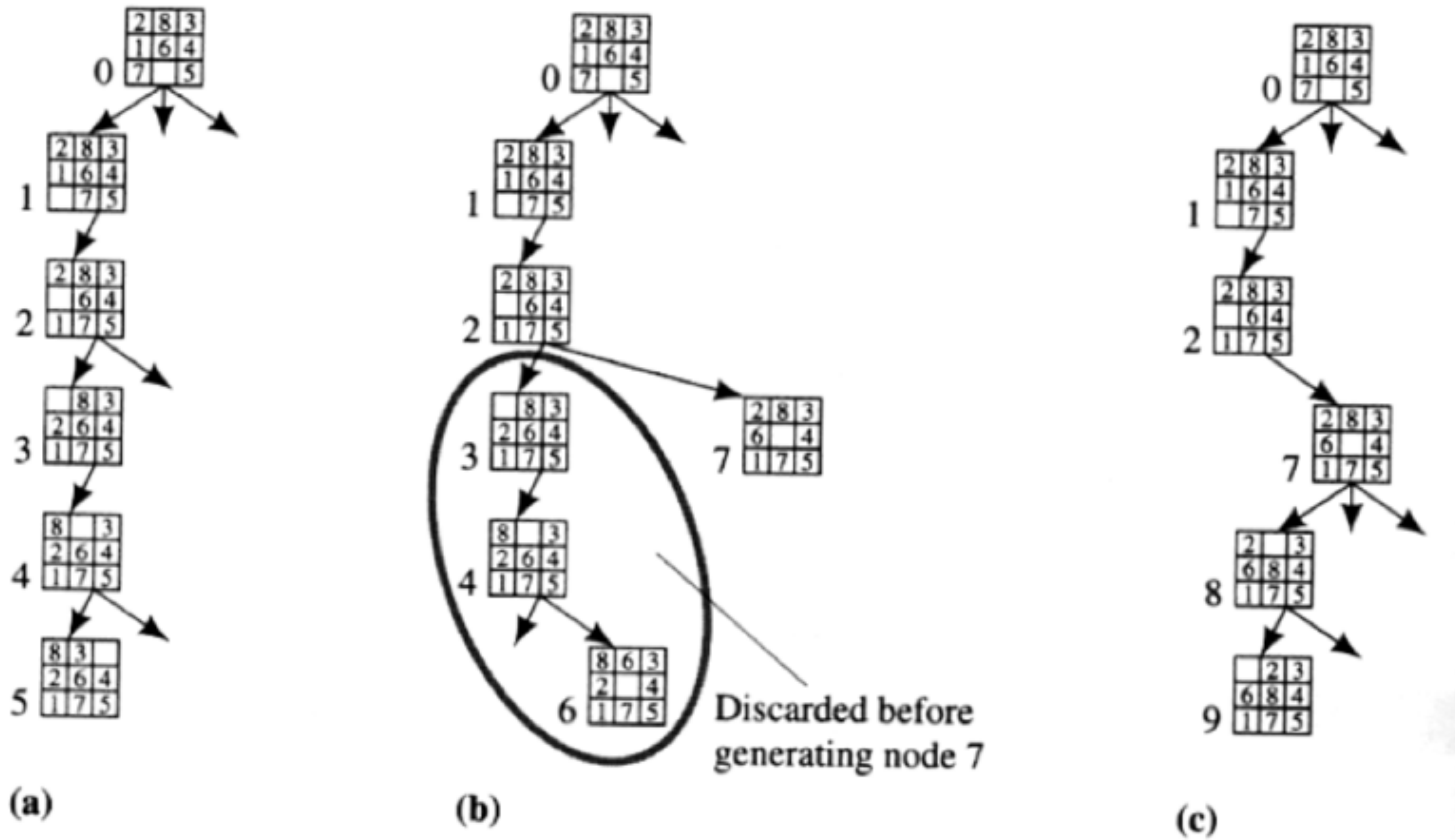
- Good:
  - Since we don't expand all nodes at a level, space complexity is modest.
  - For branching factor b and depth m, we require bm number of nodes to be stored in memory.
  - However, the worst case is still O(bm).

- Bad:
  - If you have deep search trees (or infinite -which is quite possible), DFS may end up running off to infinity and may not be able to recover.
  - Thus DFS is neither optimal nor complete

Depth-limited search

- Depth-Limited Search Modifies DFS to avoid its pitfalls.
  - Say that within a given area, we had to find the shortest path to visit 10 cities. If we start at one of the cities, then there are at least 9 other cities to visit. So 9 is the limit we impose.
- Since we impose a limit, there are little changes from DFS — with the exception that we will avoid searching an infinite path.
- DLS is complete if the limit we impose is greater than or equal to the depth of our solution.

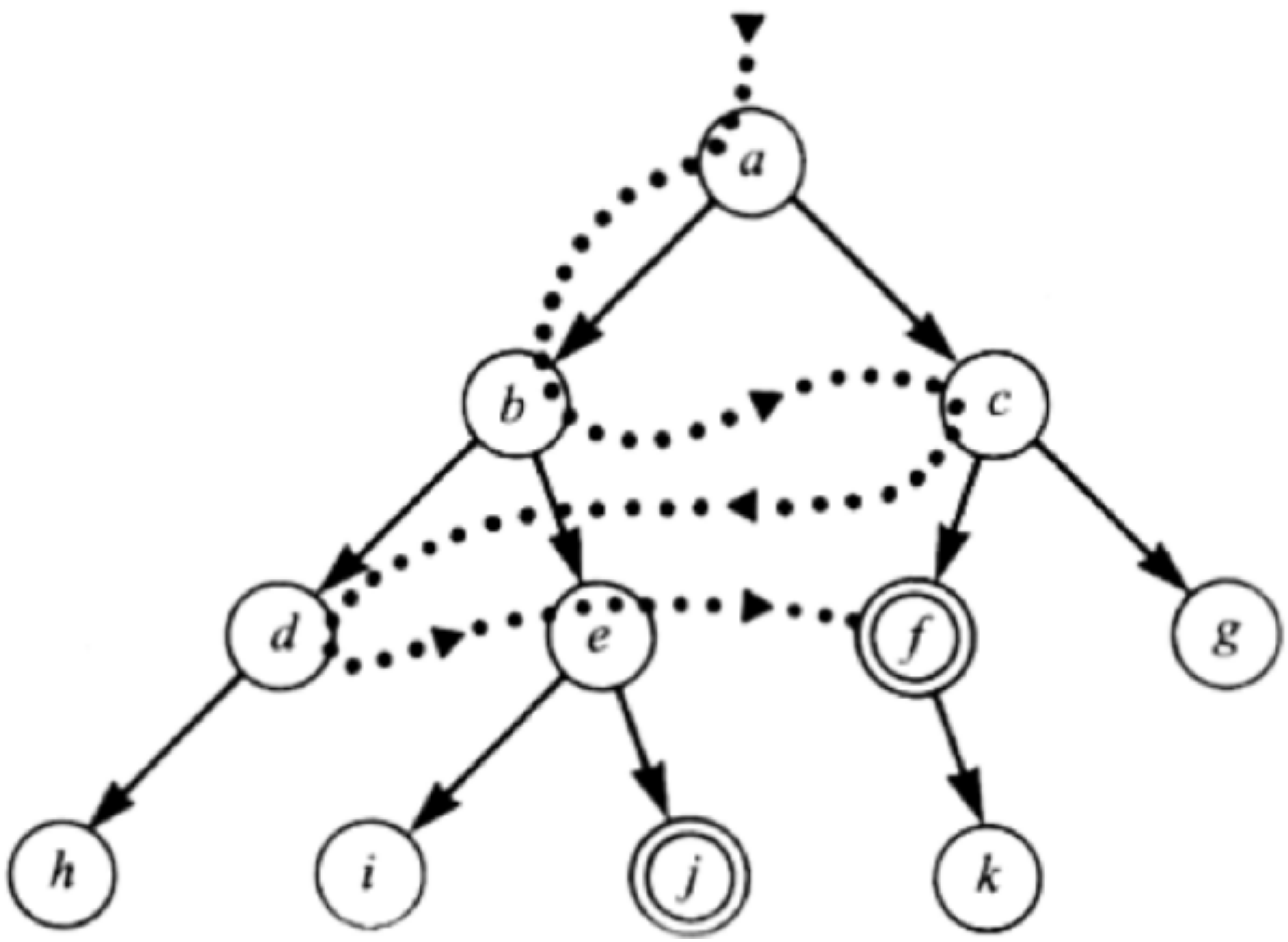
Depth-first search: eight puzzle



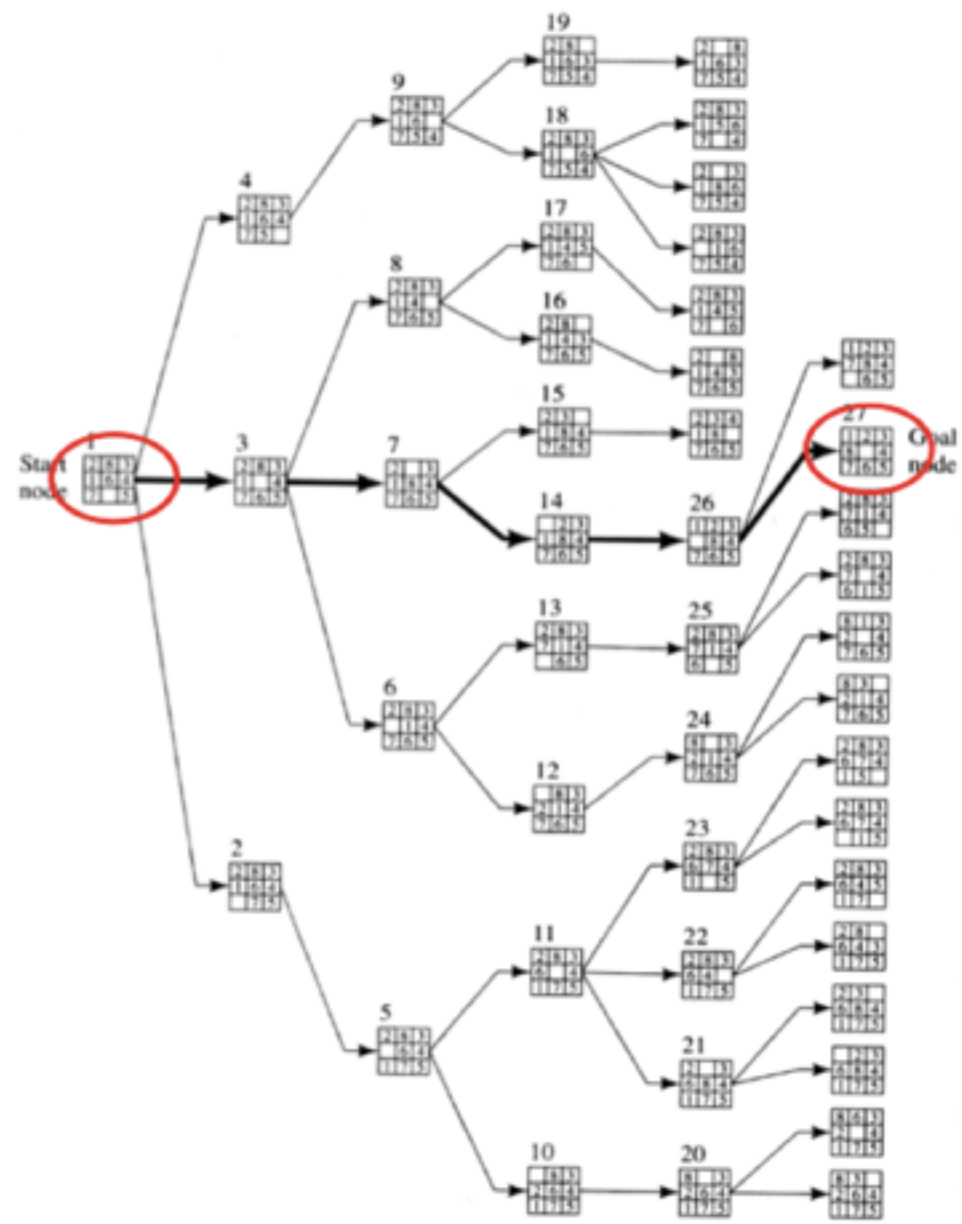
Depth-limited search: space and time complexity

- DLS is O(b^l) in time, where l is the limit we impose.
- Space complexity is bl. Space complexity is bl.
- Not optimal

Breadth-first search



Breadth-first search: example



Breadth-first search: time and space complexity

- If we look at how BFS expands from the root we see that it first expands on a fixed number of nodes, say b.
- On the second level we expand b^2 nodes.
- On the third level we expand b^3 nodes. - And so on, until it reaches bd for some depth d.

  1+ b + b^2 + b^3 + ... + b^d , which is O(b^d)

- Since all leaf nodes need to be stored in memory, space complexity is the same as time complexity.

Breadth search times

Depth	Nodes	Time	Memory
2	1100	.11 seconds	1 megabyte
4	111,100	11 seconds	106 megabytes
6	10^7	19 minutes	10 gigabytes
8	10^9	31 hours	1 terabytes
10	10^{11}	129 days	101 terabytes
12	10^{13}	35 years	10 petabytes
14	10^{15}	3,523 years	1 exabyte

Figure 3.11 Time and memory requirements for breadth-first search. The numbers shown assume branching factor $b = 10$; 10,000 nodes/second; 1000 bytes/node.

Categories of search

- Uninformed Search
  - We can distinguish the goal state(s) from the non-goal state.
  - The path and cost to find the goal is unknown.
  - Also known as blind search.

- Informed search
  - We know something about the nature of our path that might increase the efectiveness of our search
  - Generally superior to uninformed search.

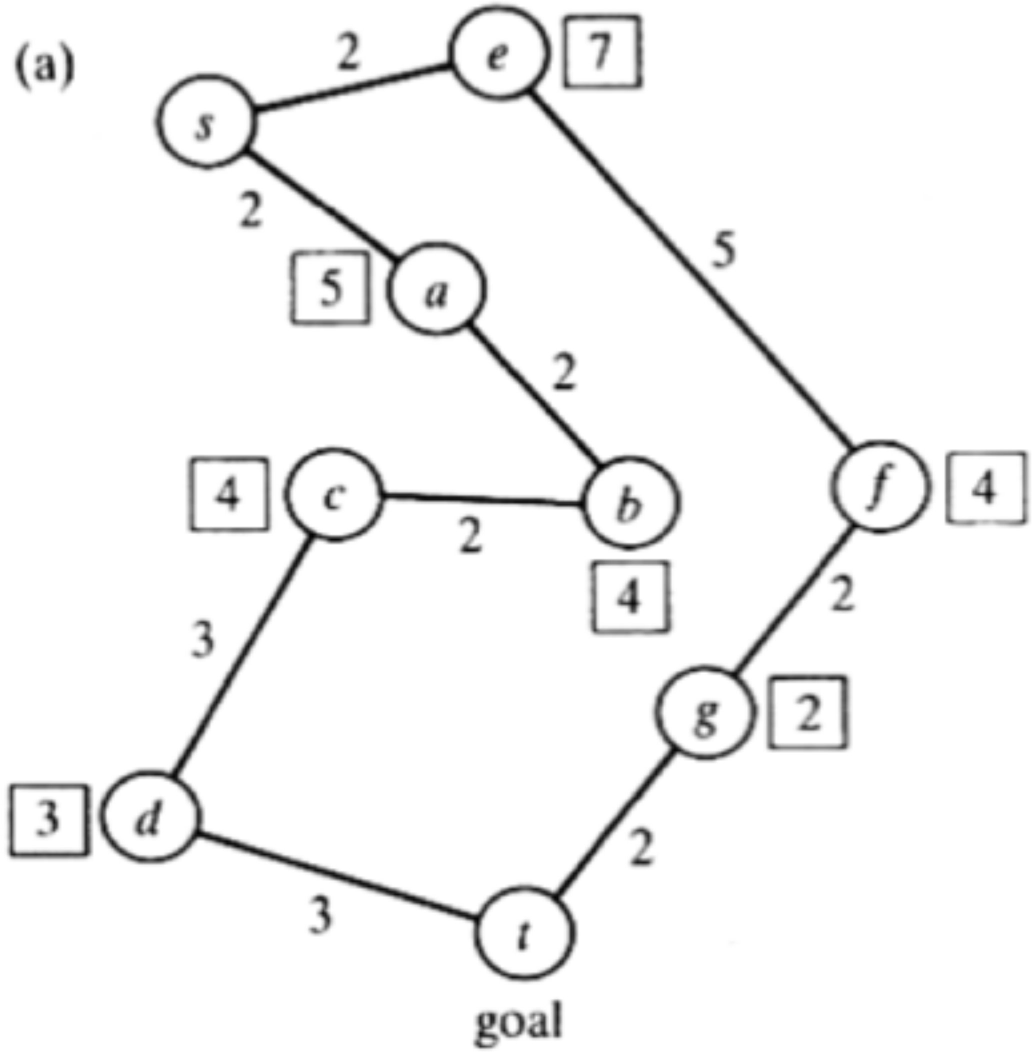
Some uninformed search strategies

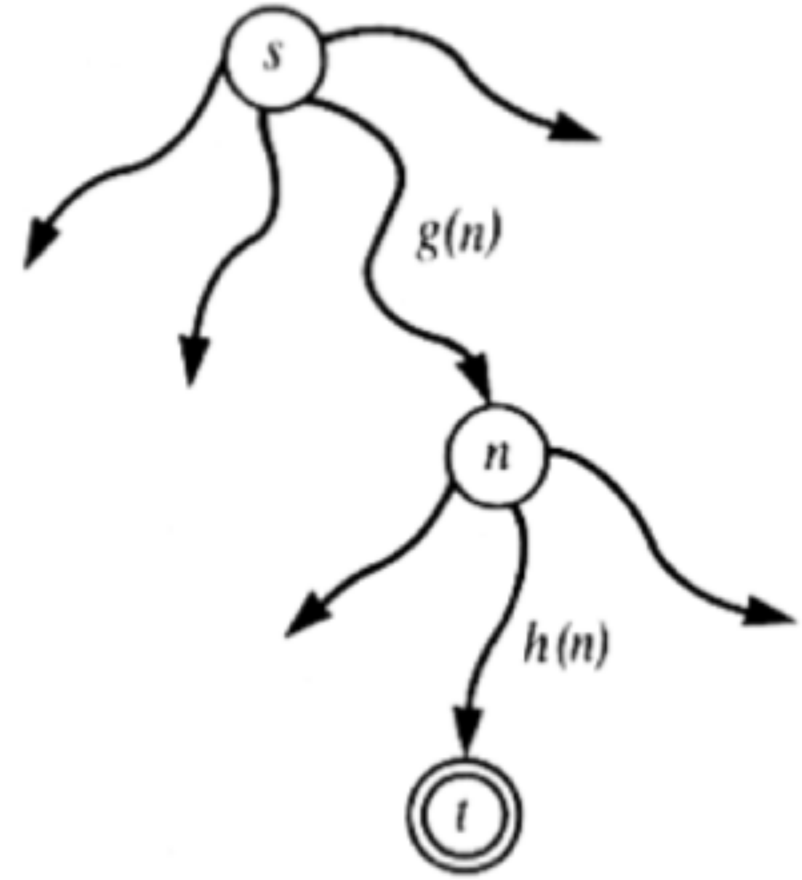
- Depth First Search
- Depth Limited Search
- Iterative Deepening Search
- Breadth First Search (complete and optimal)
- Bidirectional Search
- Uniform Cost Search

Best-first heuristics search

- Greedy Search
- Best-First Heuristic Search (A*)
  - Routing Problem
  - Best-First Search for Scheduling

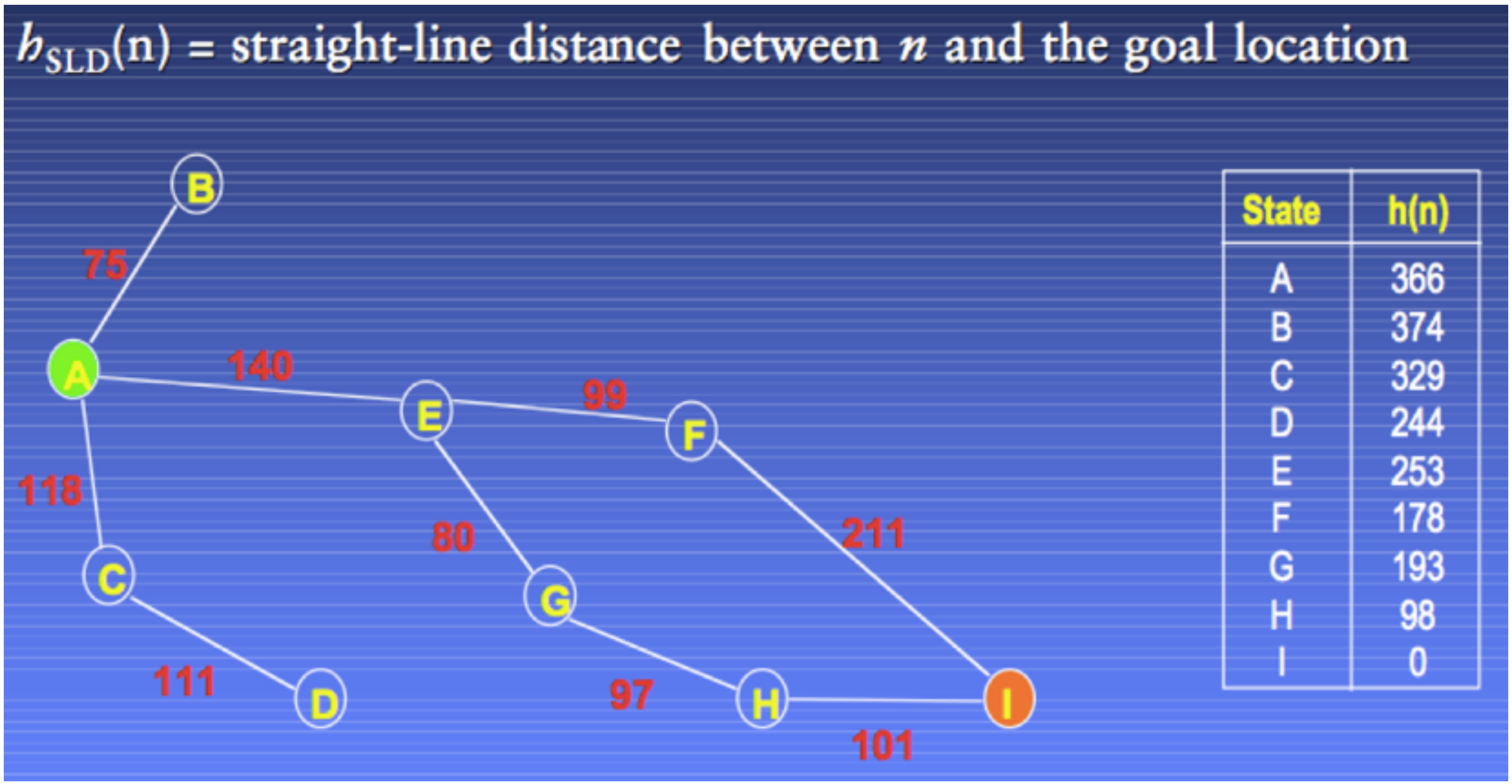
Best-first search heuristics

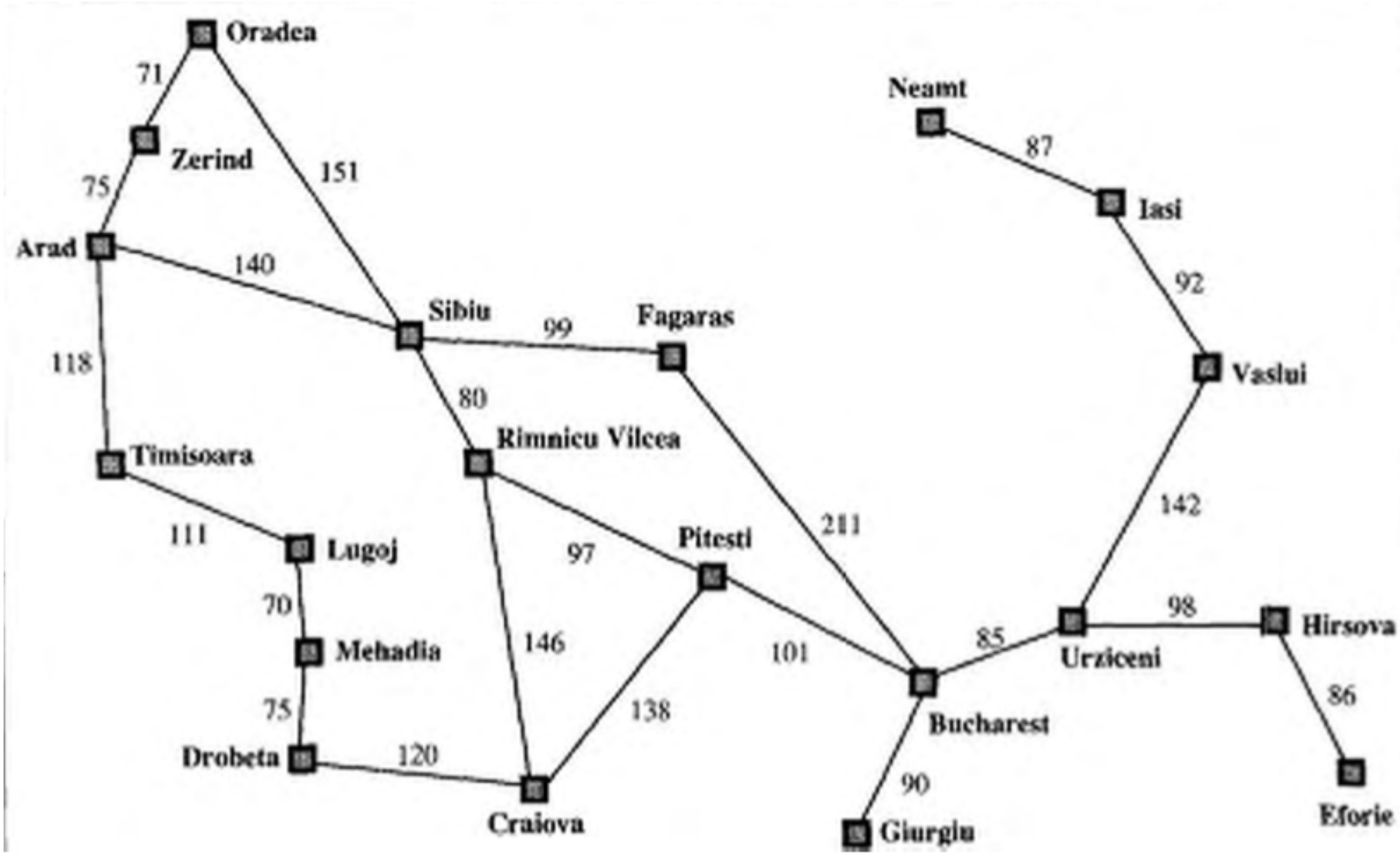




f(n) = g(n) + h(n)
f: heuristic estimator

Greedy Search (incomplete) with Straight-Line Distance Heuristic

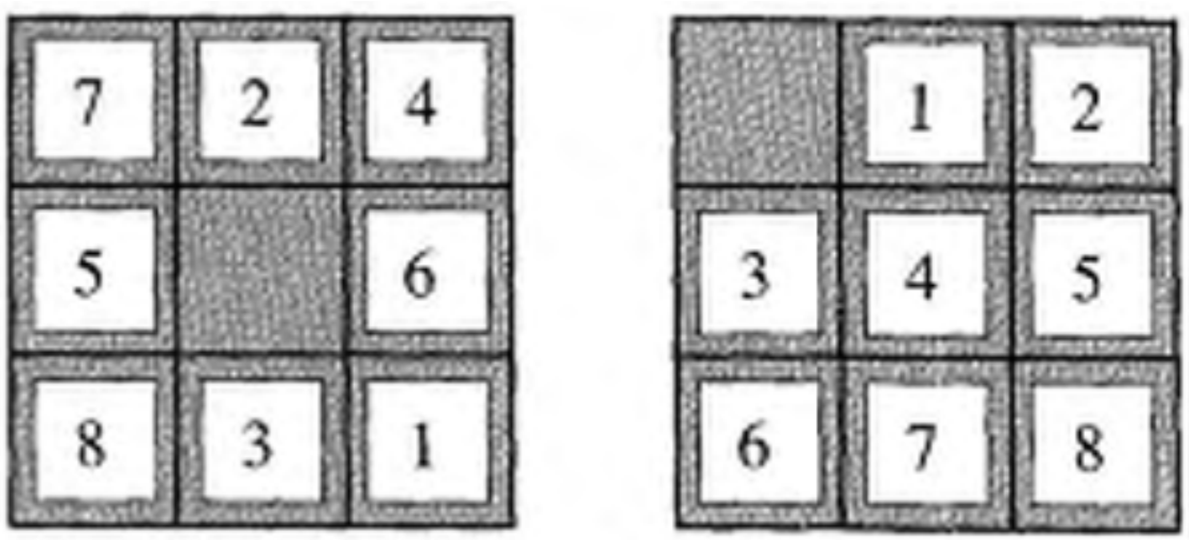



Oradea
71
Neamt
Zerind
151
87
75
Iasi
Arad
140
92
Sibiu
99
Fagaras
118
Vaslui
80
Rimnicu Vilcea
Timisoara
142
111
Lugoj
97
Pitesti
211
70
98
Mehadia
146
101
85
Urziceni
Hirsova
86
75
138
Bucharest
Drobeta
120
90
Craiova
Giurgiu
Eforie

Bucharest a --

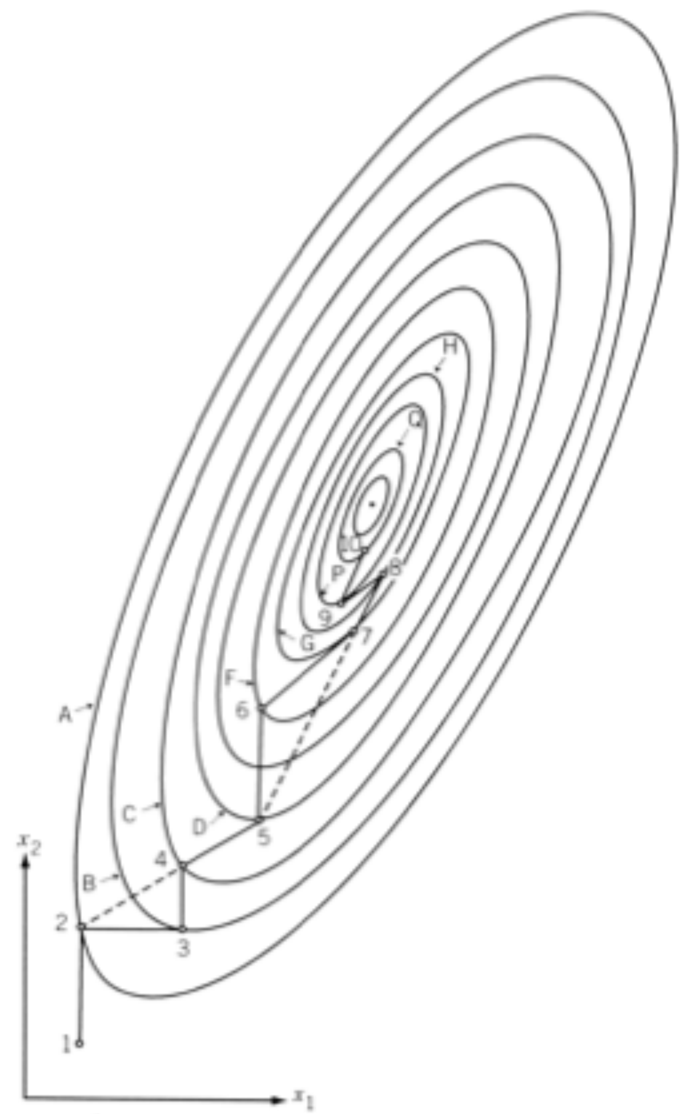
Arad	366	**Mehadia**	241
Bucharest	0	**Neamt**	234
Craiova	160	**Oradea**	380
Drobeta	242	**Pitesti**	100
Eforie	161	**Rimnicu Vilcea**	193
Fagaras	176	**Sibiu**	253
Giurgiu	77	**Timisoara**	329
Hirsova	151	**Urziceni**	80
Iasi	226	**Vaslui**	199
Lugoj	244	**Zerind**	374

Can A* be used with other sort of problems? What about the heuristic?



Start State Goal State

What about continuos espaces and non linear problems?



Some methods?

- Linear programming
- Non-linear programming
- Constrained vs Unconstrained
- Direct search vs Indirect search (descent).
- Geometric programming
- Dynamic programming
- Integer programming
- Stochastic programming
- Different sort of optimality

Espacio de aptitudes