Artificial Intelligence DT8012

Informed search

Chapter 4, AIMA 2nd ed

Chapter 3, AIMA 3rd ed

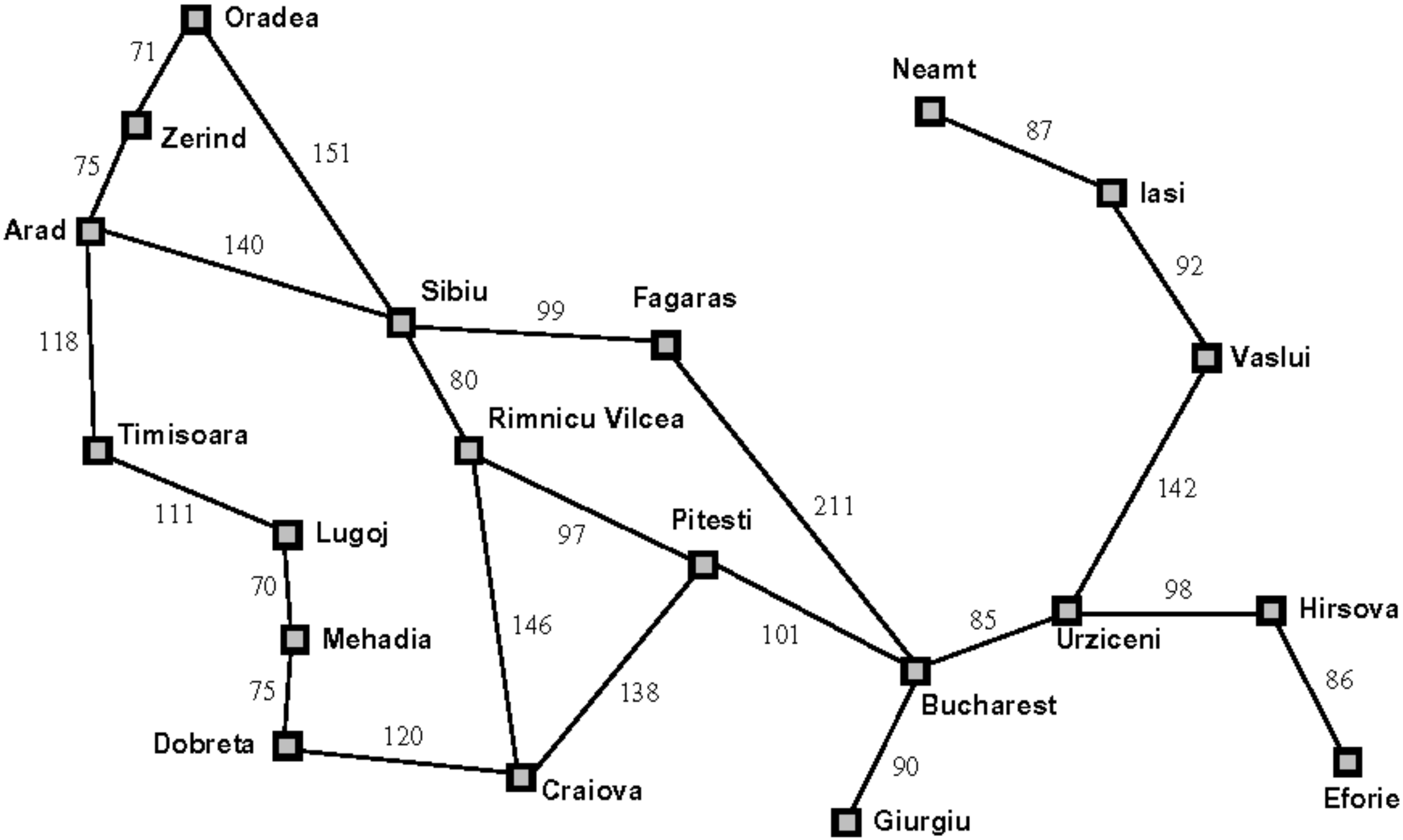

Greenland Sea
Denmark Strait
Norwegian Sea
Arctic Circle
Reykjavík
ICELAND
Hammerfest
Tromsø
Murmansk
Kiruna
White Sea
Arkhangel'sk
Luleå
Oulu
NORWAY
Umeå
FINLAND
Lake Onega
Trondheim
Lake Ladoga
Tórshavn
Faroe Islands (DENMARK)
SWEDEN
Gulf of Bothnia
Tampere
SHETLAND ISLANDS
Gävle
Turku
Helsinki
St. Petersburg
Gulf of Finland
Bergen
Oslo
ALAND ISLANDS
RUSSIA
Rockall (U.K.)
Stockholm
Tallinn
ESTONIA
ORKNEY ISLANDS
HEBRIDES
Stavanger
Moscow
North Atlantic Ocean
Aberdeen
Göteborg
Gotland
LATVIA
Riga
North Sea
Baltic Sea
Glasgow
Edinburgh
DENMARK
Öland
LITHUANIA
Vitsyebsk
Smolensk
UNITED KINGDOM
Belfast
Copenhagen
Malmö
Vilnius
Mahilyow
Kaliningrad
RUSSIA
Minsk
Isle of Man (U.K.)
Irish Sea
Bornholm
BELARUS
Dublin
Leeds
Manchester
Liverpool
Gdańsk
Hrodna
Homyel'
IRELAND
Hamburg
Brest
KINGDOM
Bremen
Berlin
Warsaw
Cardiff
Birmingham
Amsterdam
NETH.
Poznań
Kiev
London
Rotterdam
POLAND
Łódź
Rivne
Celtic Sea
Essen
Leipzig
Cologne
Wrocław
Brussels
GERMANY
UKRAINE
English Channel
Lille
BEL.
Bonn
Prague
Kraków
L'viv
Guernsey (U.K.)
Jersey (U.K.)
Frankfurt am Main
CZECH REPUBLIC
Luxembourg
Brno
Chernivtsi
SLOVAKIA
Paris
LUX.
Mykolayiv
Chişinău
Stuttgart
Bratislava
Strasbourg
Munich
Vienna
Iaşi
Nantes
LIECH.
Budapest
Cluj-Napoca
MOLDOVA
Odesa
Zürich
AUSTRIA
Bern
Vaduz
HUNGARY
ROMANIA
Bay of Biscay
FRANCE
SWITZ.
Geneva
Ljubljana
Lyon
Milan
Venice
SLOVENIA
Zagreb
Bucharest
Constanţa
Black Sea
Turin
Bordeaux
Genoa
CROATIA
BOSNIA AND HERZEGOVINA
Belgrade
Varna
Bilbao
Serbia
Toulouse
SAN MARINO
Florence
MONACO
Sarajevo
BULGARIA
Andorra la Vella
Marseille
Ligurian Sea
Montenegro
Sofia
Porto
ITALY
Adriatic Sea
İstanbul
ANDORRA
Podgorica
Skopje
F.Y.R.O.M.
Zaragoza
Corsica (FRANCE)
Rome
PORTUGAL
Madrid
Barcelona
VATICAN CITY
Tiranë
Thessaloníki
Bursa
ALB.
TURKEY
Lisbon
SPAIN
Naples
Valencia
Balearic Sea
Tyrrhenian Sea
Sardinia
GREECE
İzmir
Aegean Sea
BALEARIC ISLANDS
Athens
Sevilla
Cagliari
Ionian Sea
Mediterranean Sea
Palermo
Gibraltar (U.K.)
Málaga
Sicily
Rhodes
Strait of Gibraltar
Ceuta (SPAIN)
Alborán Sea
Algiers
Crete
Melilla (SPAIN)
Oran
Tunis
Scale 1: 19,500,000
Valletta
MALTA
Rabat
Casablanca
MOROCCO
ALGERIA
TUNISIA
300 Kilometers
300 Miles

Romania

SATU MARE
BAIA MARE
SUCEAVA
BOTOȘANI
ZALĂU
IAȘI
ORADEA
BISTRIȚA
PIATRA - NEAMȚ
CLUJ NAPOCA
TÎRGU MUREȘ
VASLUI
BACĂU
MIERCUREA - CIUC
ARAD
ALBA IULIA
SFÎNTU GHEORGHE
TIMIȘOARA
DEVA
SIBIU
FOCȘANI
BRAȘOV
GALAȚI
REȘIȚA
BRĂILA
RÎMNICU VÎLCEA
BUZĂU
TULCEA
PLOIEȘTI
TÎRGOVIȘTE
PITEȘTI
DROBETA TURNU SEVERIN
SLOBOZIA
BUCUREȘTI
SLATINA
CRAIOVA
CONSTANȚA
CĂLĂRAȘI
ALEXANDRIA
THE BLA CK SEA
GIURGIU

Romania

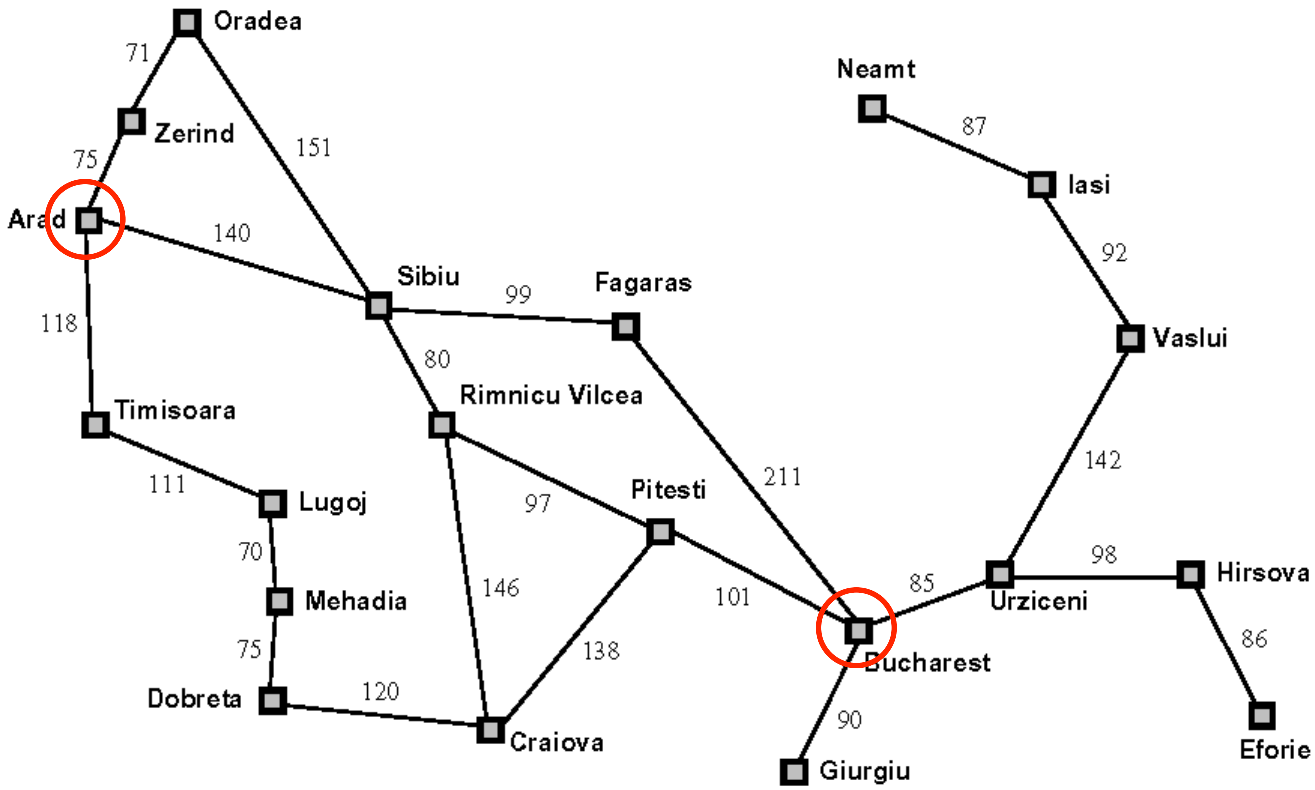
SATU MARE
BAIA MARE
SUCEAVA
BOTOŞANI
ZALĂU
IAŞI
BISTRIŢA
PIATRA - NEAMŢ
CLUJ NAPOCA
TÎRGU MUREŞ
BACĂU
MIERCUREA - CIUC
ALBA IULIA
SFÎNTU GHEORGHE
SIBIU
DEVA
FOCŞANI
BRAŞOV
GALAŢI
REŞIŢA
BRĂILA
BUZĂU
TULCEA
PLOIEŞTI
TÎRGOVIŞTE
DROBETA TURNU SEVERIN
SLOBOZIA
BUCUREŞTI
SLATINA
CĂLĂRAŞI
ALEXANDRIA
GIURGIU
THE BLACK SEA


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

Romania problem

Initial state: Arad

Find the minimum distance path to Bucharest.



Informed search

Searching for the goal and knowing something about in which direction it is.

Evaluation function: $f(n)$
- Expand the node with minimum $f(n)$

Heuristic function: $h(n)$
- Our estimated cost of the path from node n to the goal.

Example heuristic function *h(n)*

Arad	366	**Mehadia**	241
Bucharest	0	**Neamt**	234
Craiova	160	**Oradea**	380
Dobreta	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

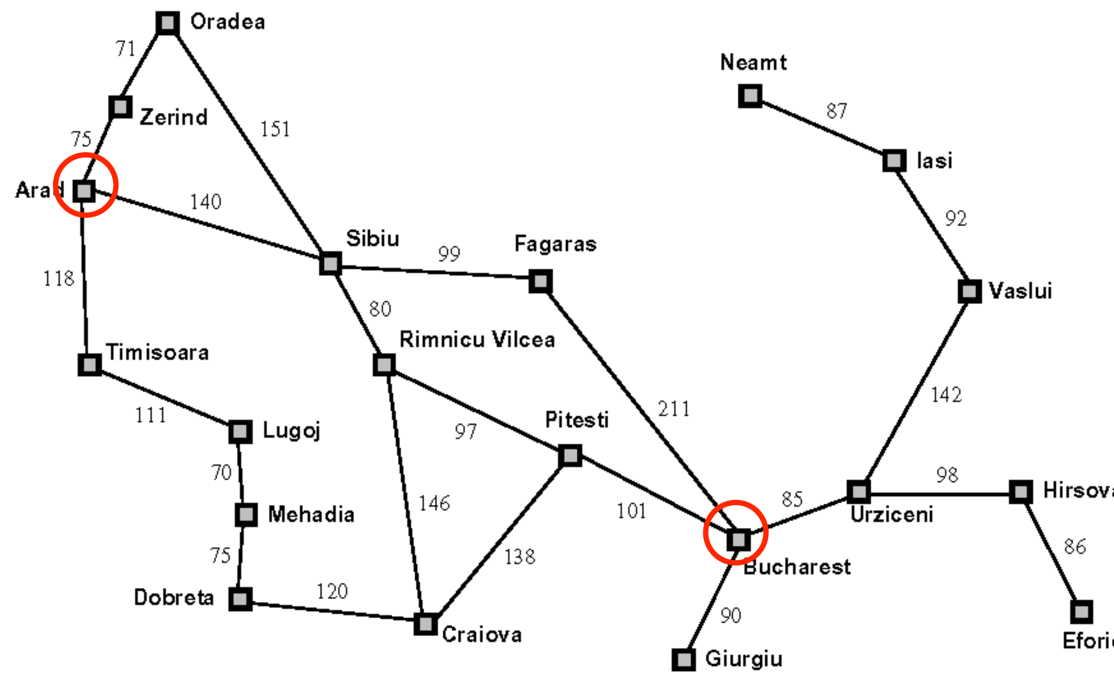
h_{SLD} – Straight-line distances (km) to Bucharest

Greedy best-first (GBFS)

Expand the node that appears to be closest to the goal: $f(n) = h(n)$

- Incomplete (infinite paths, loops)
- Not optimal (unless the heuristic function is a correct estimate)
- Space and time complexity ~ $O(b^d)$

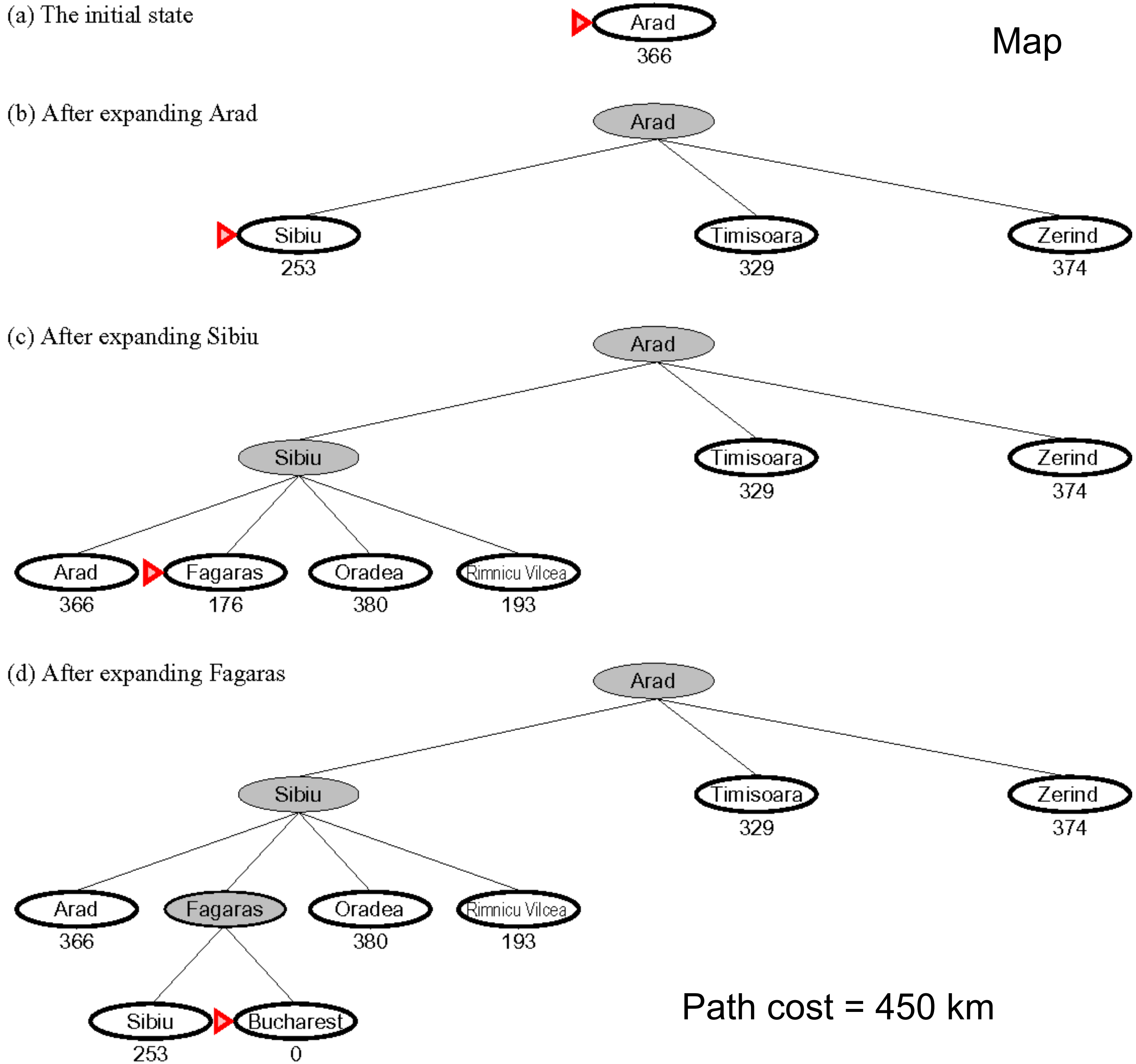
Assignment: Expand the nodes in the greedy-best-first order, beginning from Arad and going to Bucharest



Arad	366	**Mehadia**	241
Bucharest	0	**Neamt**	234
Craiova	160	**Oradea**	380
Dobreta	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

These are the h(n) values.

Map

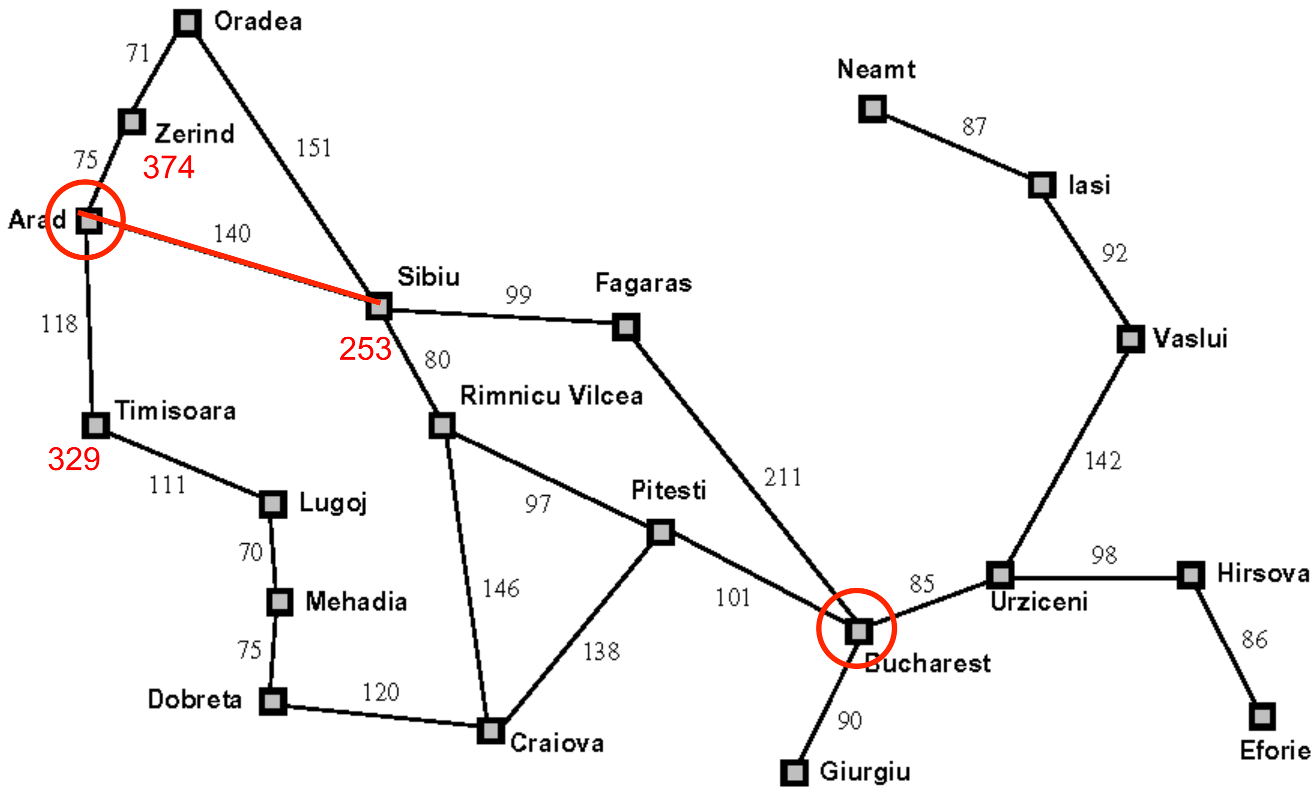


Path cost = 450 km

Romania problem: GBFS

Initial state: Arad

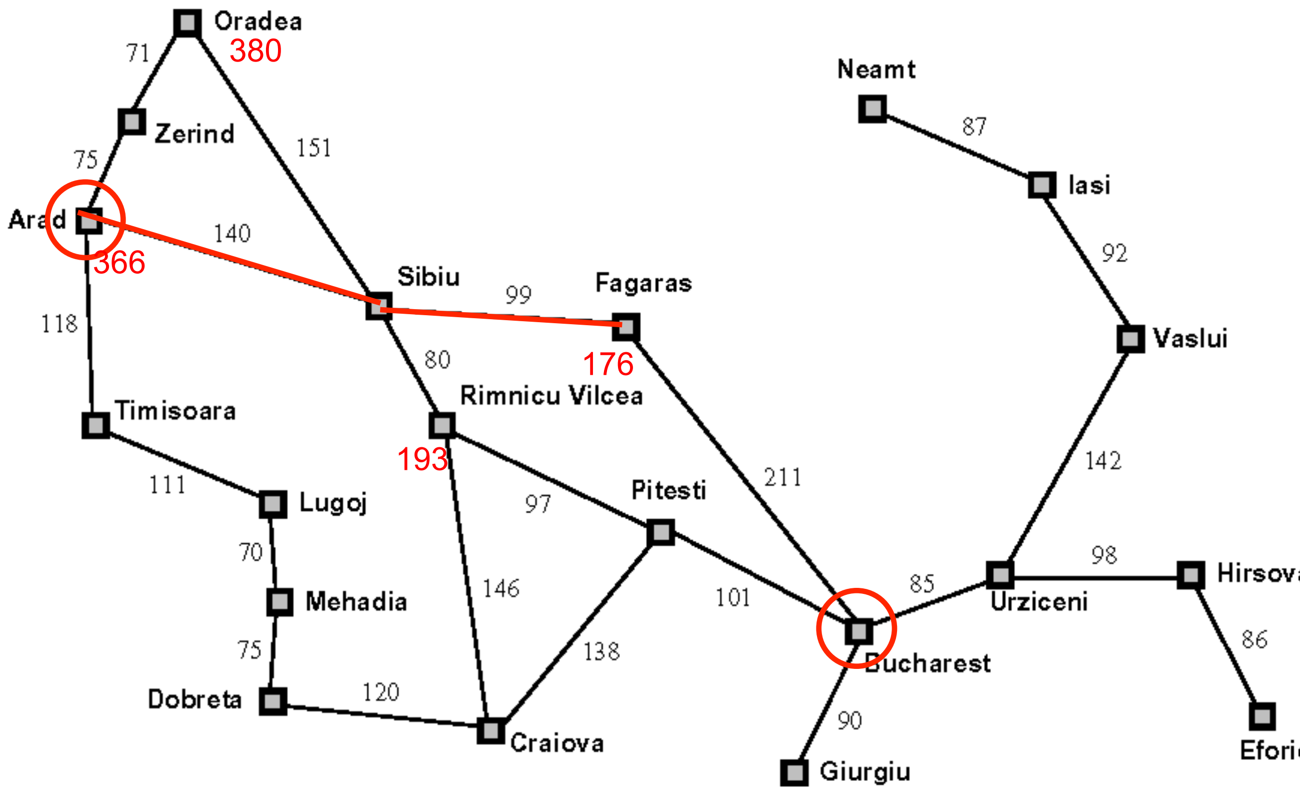
Find the minimum distance path to Bucharest.



Romania problem: GBFS

Initial state: Arad

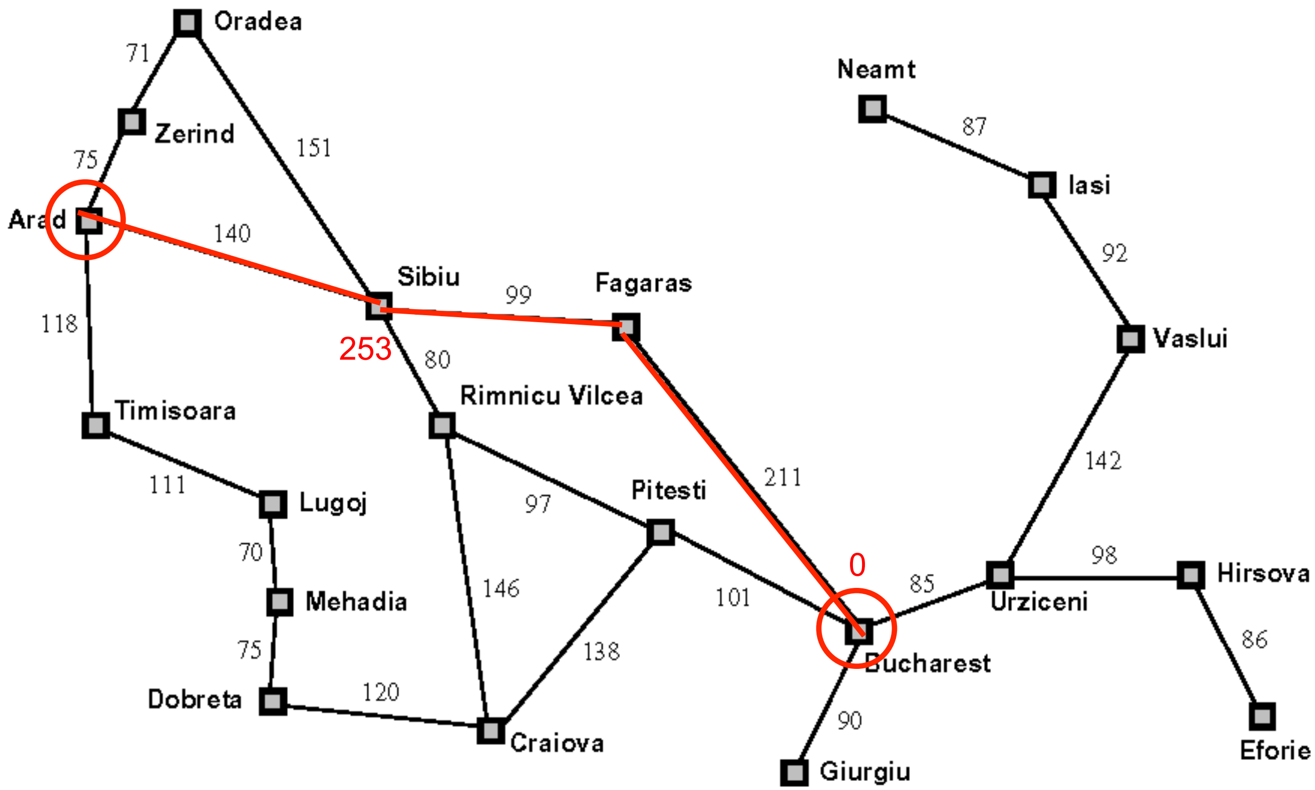
Find the minimum distance path to Bucharest.



Romania problem: GBFS

Initial state: Arad

Find the minimum distance path to Bucharest.



Not the optimal solution

Path cost = 450 km

A and A* best-first search

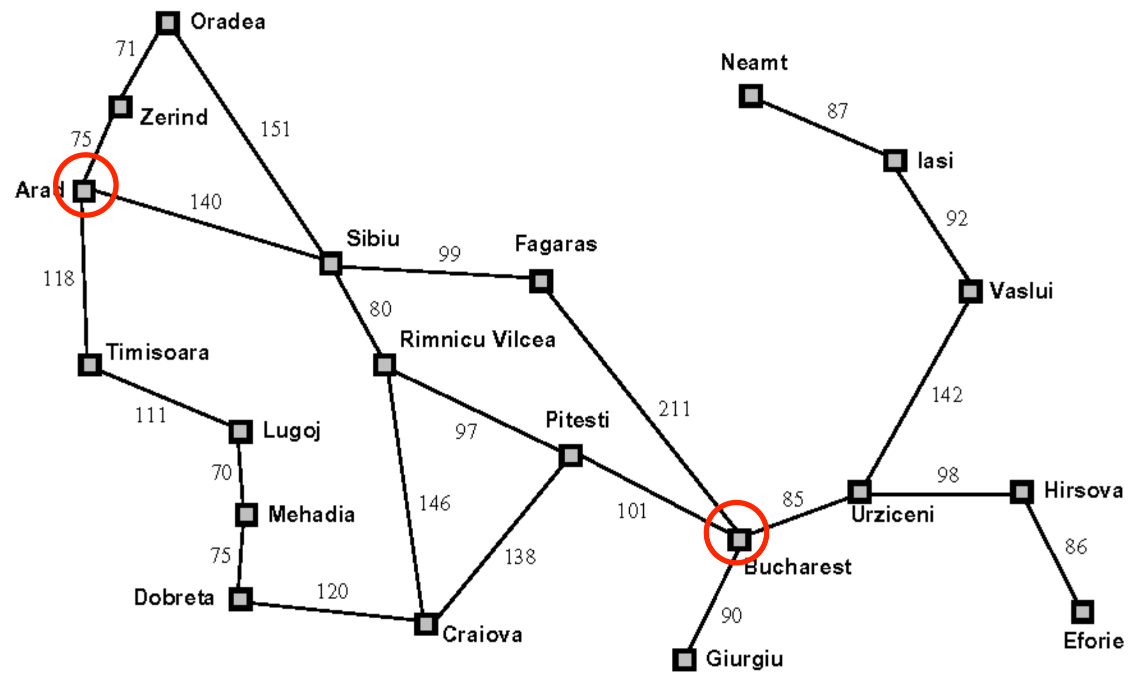
A: Improve greedy search by discouraging wandering off: $f(n) = g(n) + h(n)$

Here $g(n)$ is the cost to get to node n from the start position.

This penalizes taking steps that don't improve things considerably.

A*: Use an *admissible* heuristic, i.e. a heuristic $h(n)$ that never overestimates the true cost for reaching the goal from node n.

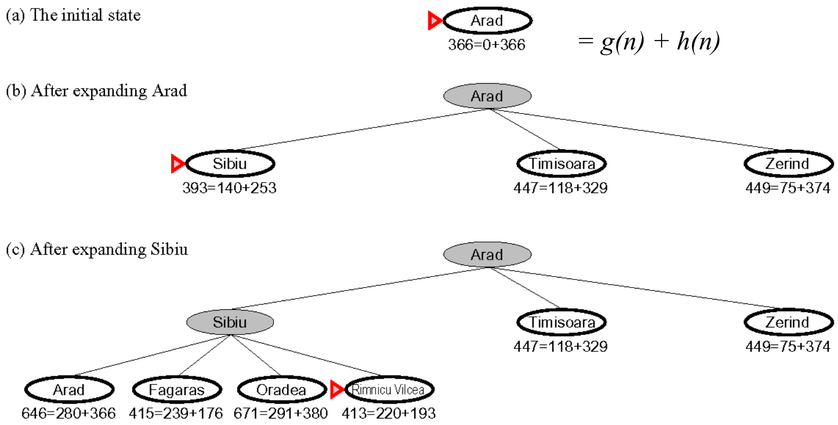
Assignment: Expand the nodes in the A* order, beginning from Arad and going to Bucharest



Arad	366	**Mehadia**	241
Bucharest	0	**Neamt**	234
Craiova	160	**Oradea**	380
Dobreta	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

These are the g(n) values.

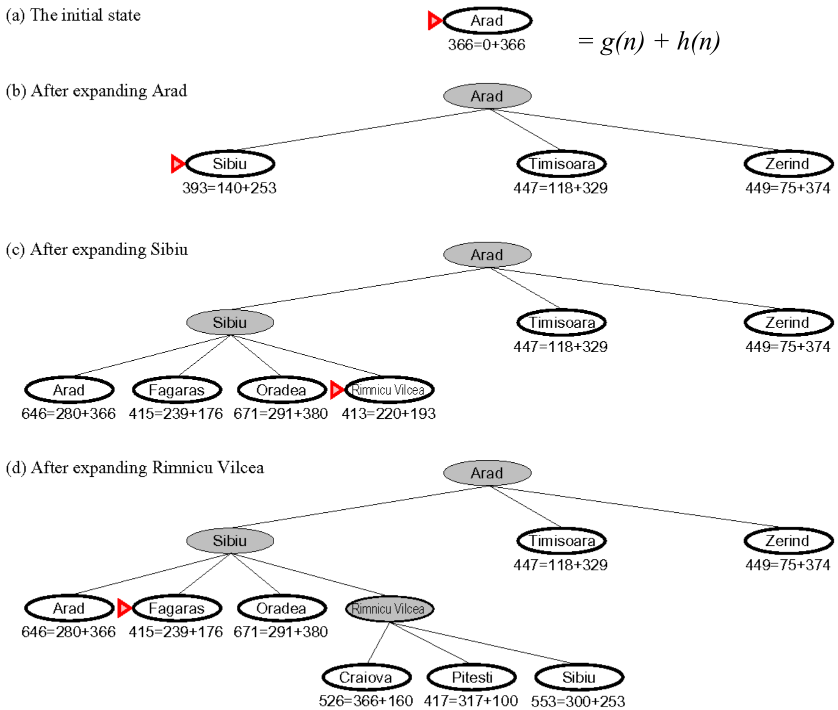
These are the h(n) values.



The straight-line distance never overestimates the true distance; it is an admissible heuristic.

A* on the Romania problem.

Rimnicu-Vilcea is expanded before Fagaras.

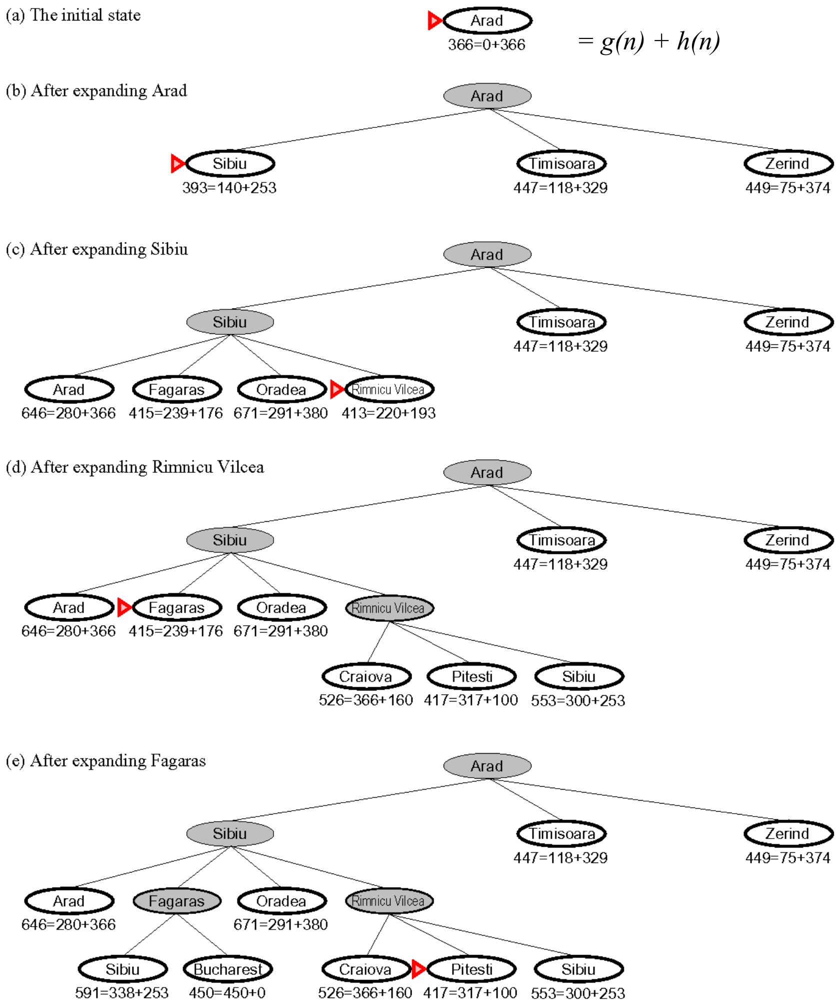


The straight-line distance never overestimates the true distance; it is an admissible heuristic.

A* on the Romania problem.

Rimnicu-Vilcea is expanded before Fagaras.

The gain from expanding Rimnicu-Vilcea is too small so the A* algorithm backs up and expands Fagaras.



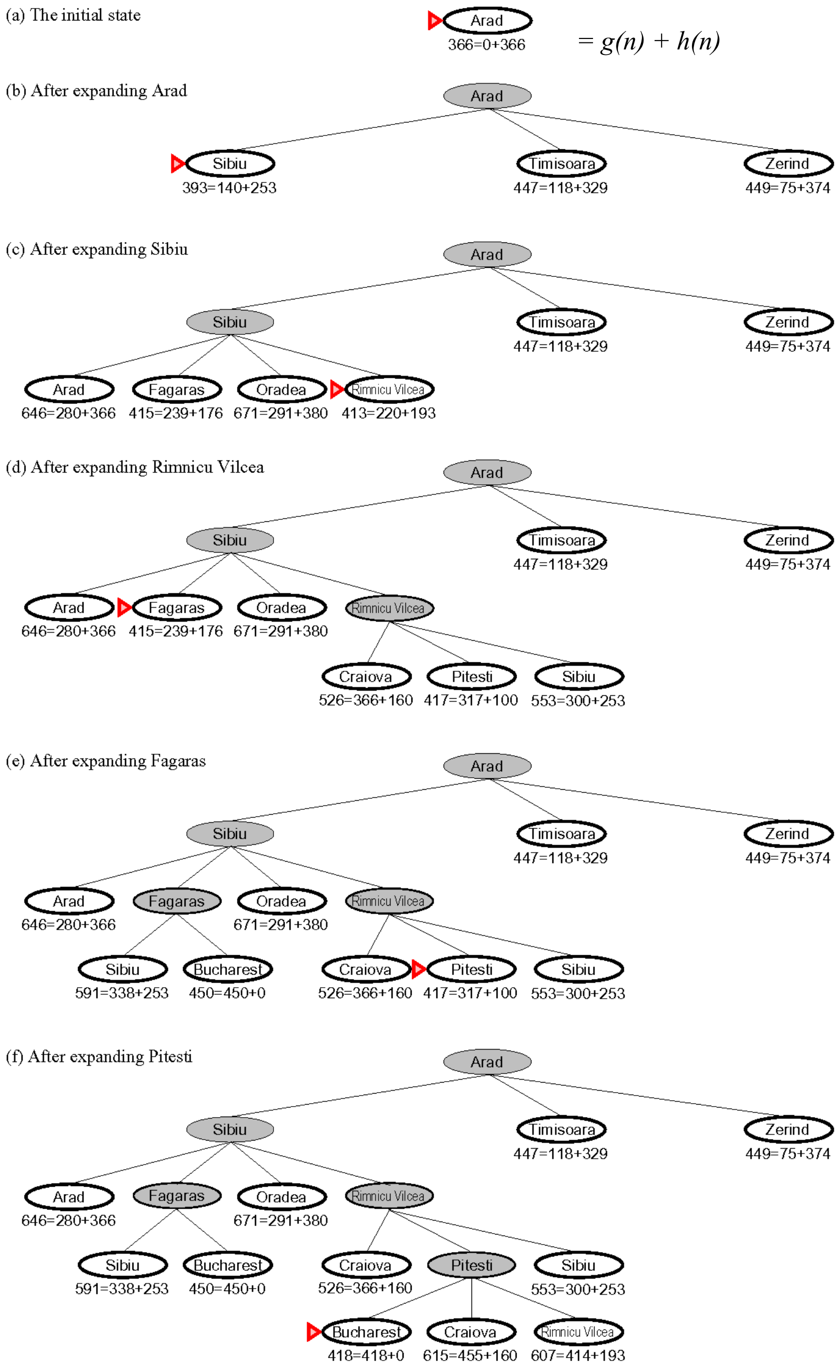
The straight-line distance never overestimates the true distance; it is an admissible heuristic.

A* on the Romania problem.

Rimnicu-Vilcea is expanded before Fagaras.

The gain from expanding Rimnicu-Vilcea is too small so the A* algorithm backs up and expands Fagaras.

None of the descentants of Fagaras is better than a path through Rimnicu-Vilcea; the algorithm goes back to Rimnicu-Vilcea and selects Pitesti.



The straight-line distance never overestimates the true distance; it is an admissible heuristic.

A* on the Romania problem.

Rimnicu-Vilcea is expanded before Fagaras.

The gain from expanding Rimnicu-Vilcea is too small so the A* algorithm backs up and expands Fagaras.

None of the descentants of Fagaras is better than a path through Rimnicu-Vilcea; the algorithm goes back to Rimnicu-Vilcea and selects Pitesti.

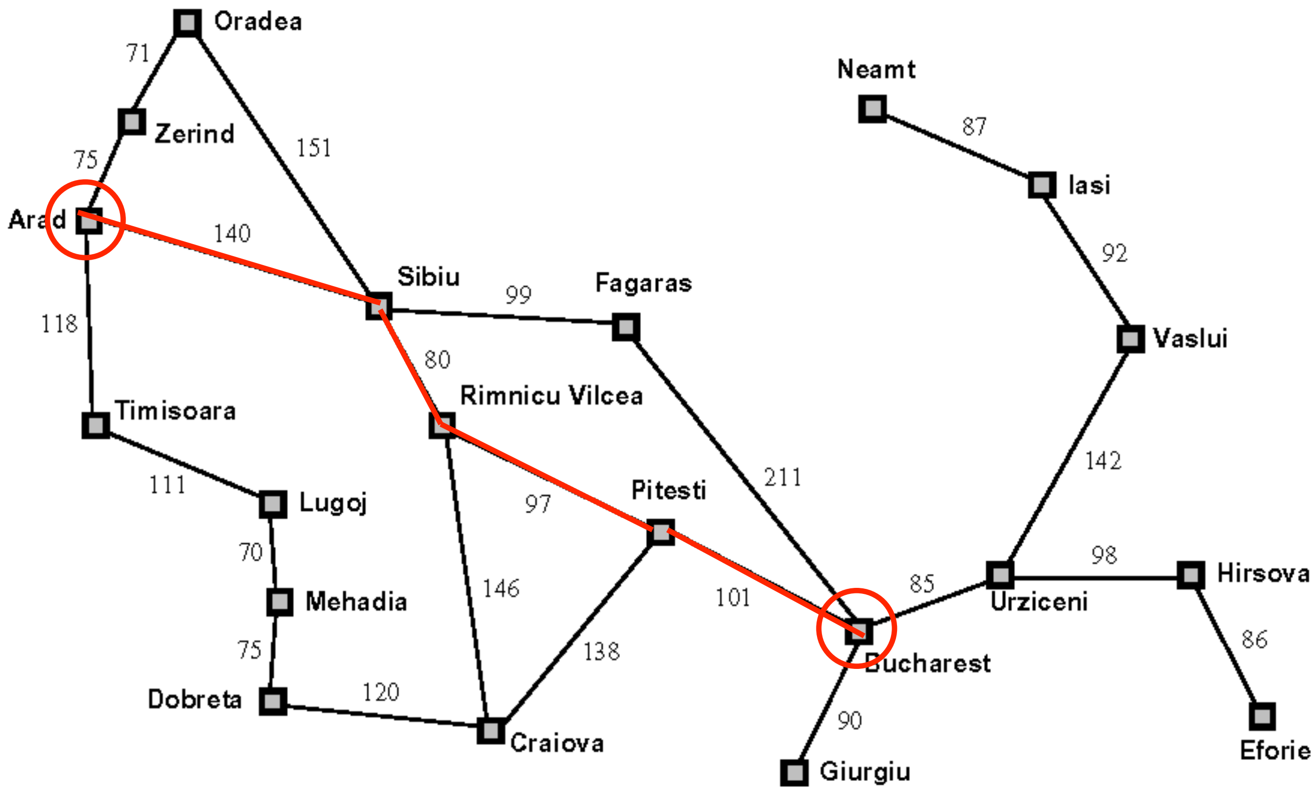
The final path cost = 418 km

This is the optimal solution.

Romania problem: A*

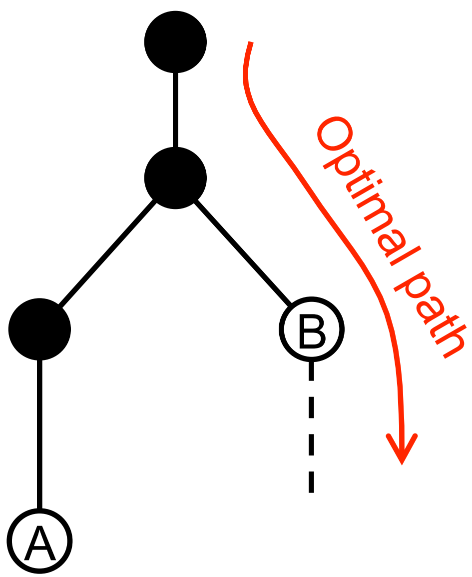
Initial state: Arad

Find the minimum distance path to Bucharest.



The optimal solution
Path cost = 418 km

Theorem: A* tree-search is optimal

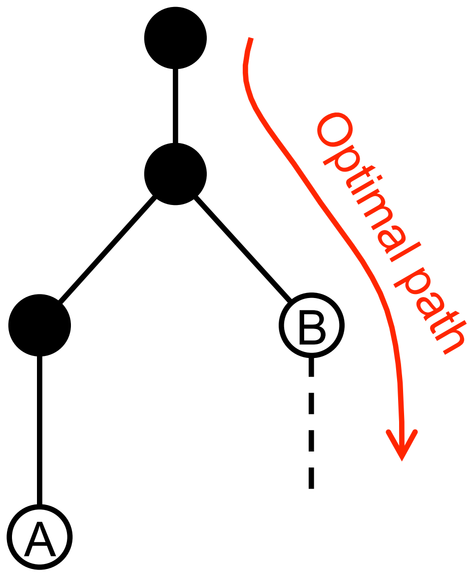


A and B are two nodes on the fringe.

A is a suboptimal goal node and B is a node on the optimal path.

Optimal path cost = C

Theorem: A* tree-search is optimal



A and B are two nodes on the fringe.

A is a <u>suboptimal</u> goal node and B is a node on the optimal path.

Optimal path cost = C

$h(A) = 0$

$$f(A) = g(A) + h(A) = g(A) > C$$

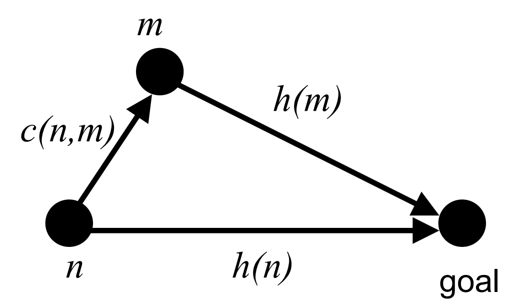
$$f(B) = g(B) + h(B) \leq C$$

$h(n)$ is admissive heuristic

Is A* graph-search optimal?

- For graph-search we add the requirement of *consistency* (monotonicity):

$$h(n) \leq c(n,m) + h(m)$$



$c(n,m)$ = step cost for going from node n to node m (n comes before m)

A* graph search with consistent heuristic is optimal

Theorem:

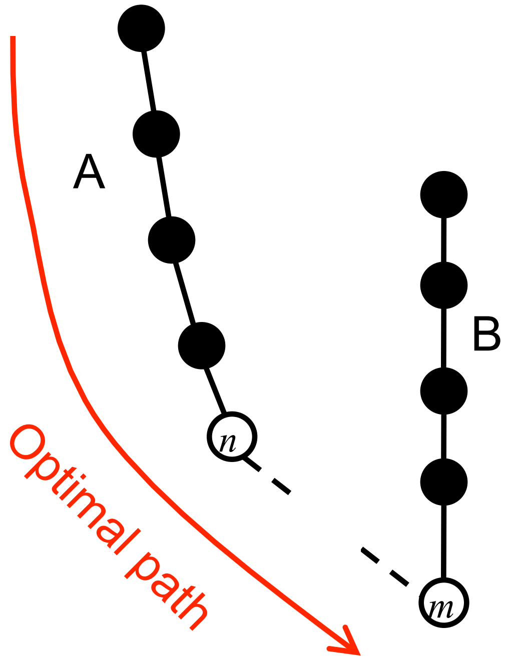
If the consistency condition on $h(n)$ is satisfied, then when A* expands a node n, it has already found an optimal path to n.

This follows from the fact that consistency means that $f(n)$ is nondecreasing along a path in the graph

$$f(m) = g(m) + h(m) = g(n) + c(n,m) + h(m) \geq g(n) + h(n) = f(n)$$

if m comes after n along a path

Proof



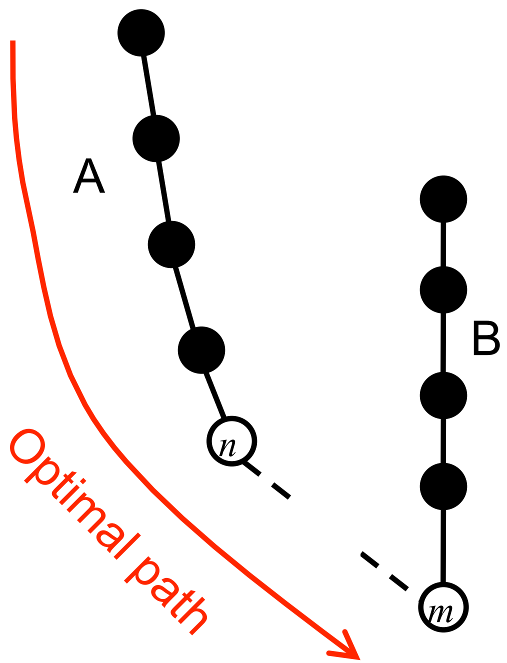
A* has reached node m along the alternative path B.

Path A is the <u>optimal</u> path to node m.

Node n precedes m along the optimal path A.

Both n and m are on the fringe and A* is about to expand m.

Proof



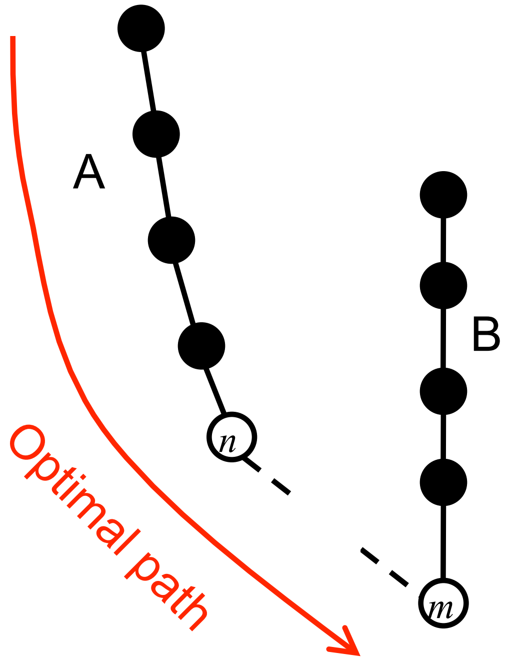
A* has reached node m along the alternative path B.

Path A is the optimal path to node m.
$\Rightarrow g_A(m) \leq g_B(m)$

Node n precedes m along the optimal path A.
$\Rightarrow f_A(n) \leq f_A(m)$

Both n and m are on the fringe and A* is about to expand m.
$\Rightarrow f_B(m) \leq f_A(n)$

Proof



$$f_B(m) = g_B(m) + h(m) \leq g_A(n) + h(n) = f_A(n)$$

$$h(n) \leq c_A(n,m) + h(m)$$

$$\Rightarrow$$

$$g_B(m) \leq g_A(n) + c_A(n,m) = g_A(m)$$

But path A is optimal to reach m why
$g_A(m) \leq g_B(m)$

Thus, either $m = n$ or contradiction.

$\Rightarrow$ A* graph-search with consistent heuristic always finds the optimal path

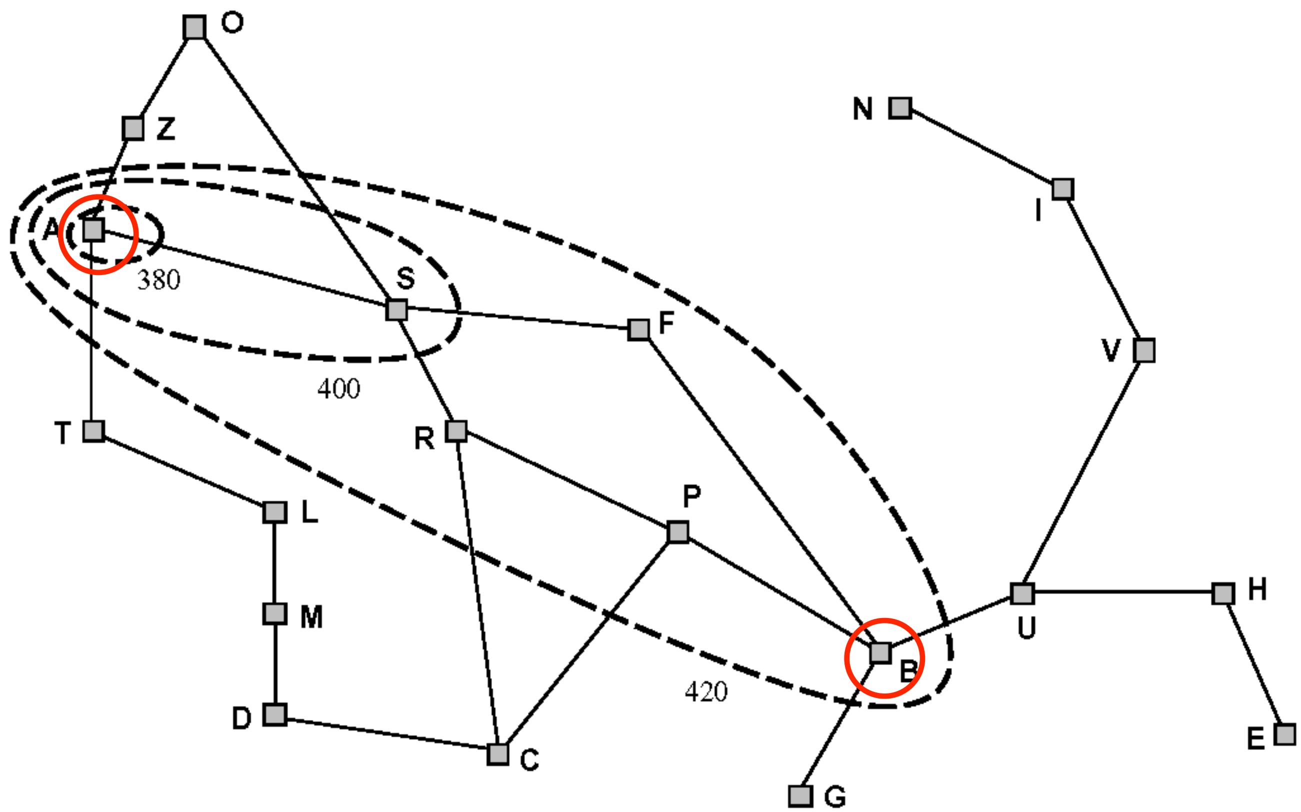
A*

- Optimal
- Complete
- Optimally efficient (no algorithm will expand fewer nodes, given the same heuristic)
- Memory requirement exponential...(bad)
- Time complexity exponential (worst case, depends on heuristic)

Romania problem: A*

Initial state: Arad

Find the minimum distance path to Bucharest.



The optimal solution
Path cost = 418 km

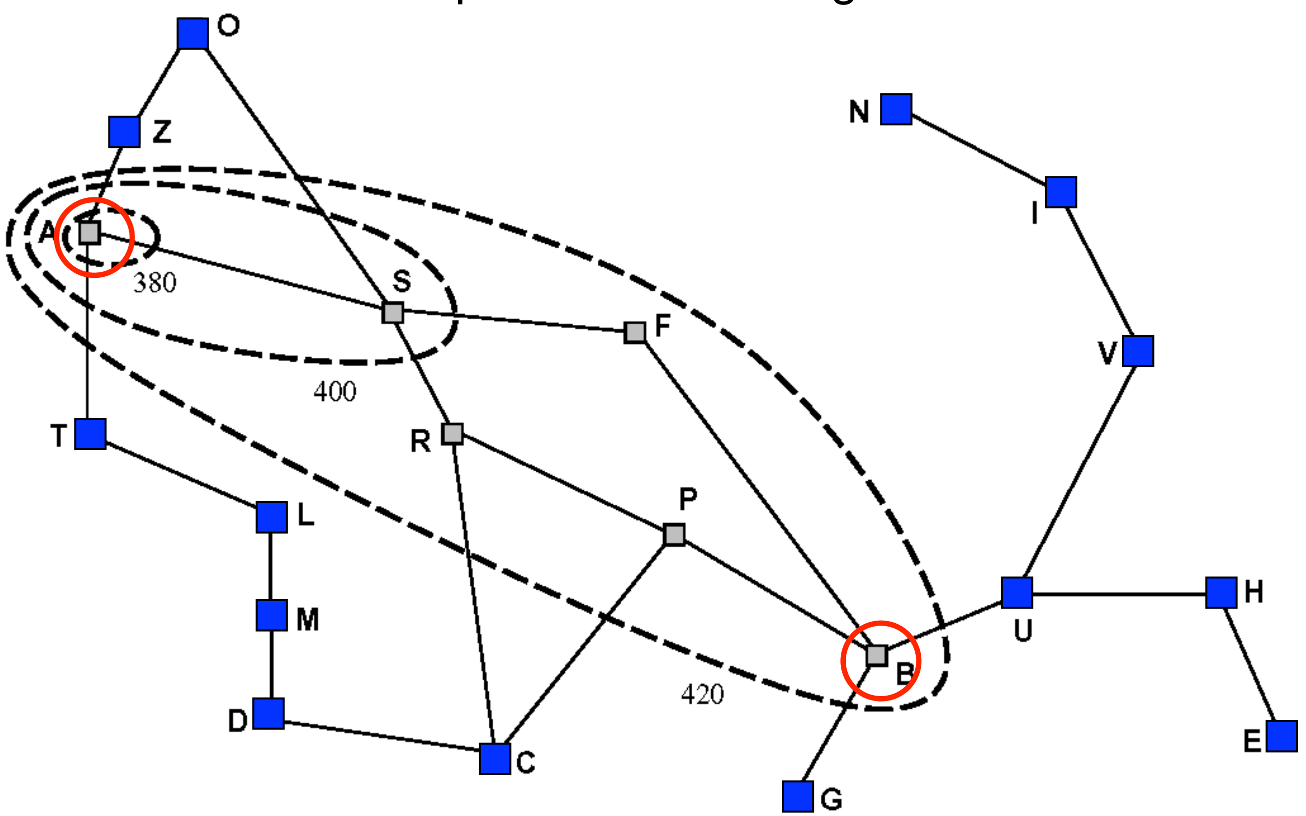
Romania problem: A*

Initial state: Arad

Find the minimum distance path to Bucharest.

Never expanded nodes

A* expands all nodes with f-value less than the optimal cost to the goal.



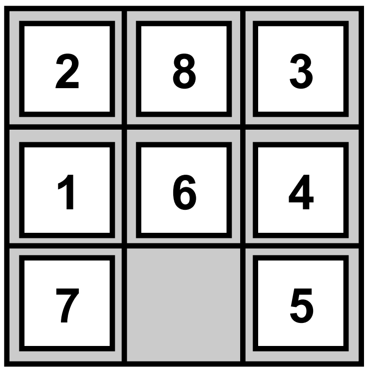
The optimal solution
Path cost = 418 km

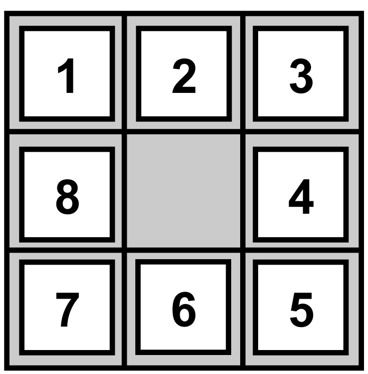
Memory bounded search

- Iterative deepening A* (IDA*) (uses *f* cost)
- Recursive best-first search (RBFS)
  - Depth-first but keep track of best *f*-value so far above.
- Memory-bounded A* (MA*/SMA*)
  - Drop old/bad nodes when memory gets full (but parent remembers worst deleted child).

Best of these is SMA*

Heuristic functions 8-puzzle



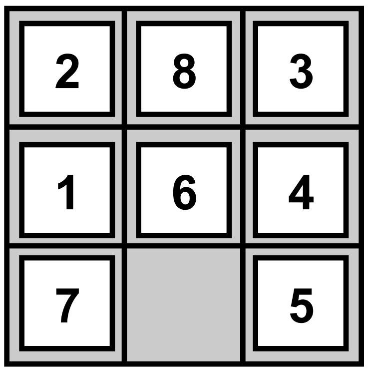


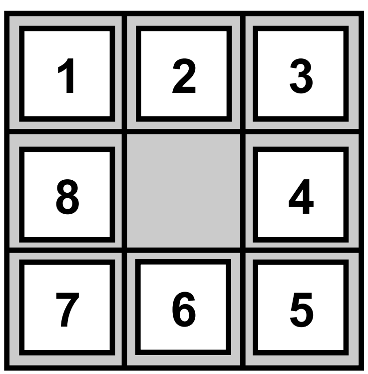
Goal state

Can you come up with heuristics for the 8-puzzle?

Think about it for a while and come with suggestions.

Heuristic functions 8-puzzle





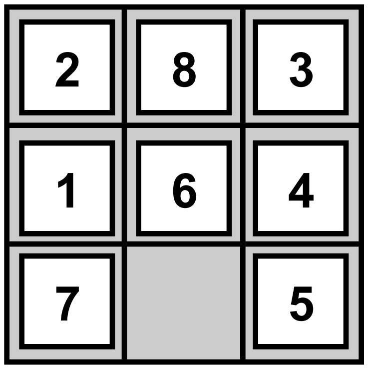
Goal state

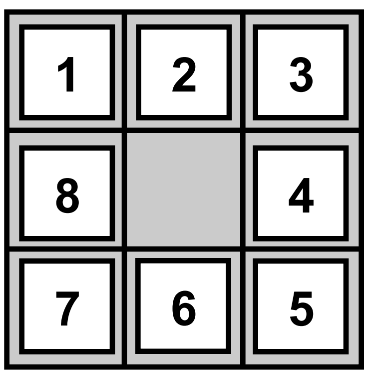
- h_1 = The number of misplaced tiles.
- h_2 = The sum of the distances of the tiles from their respective goal positions (Manhattan distance).

Both are admissive

Heuristic functions 8-puzzle

Initial state





Goal state

- h_1 = The number of misplaced tiles.

Assignment: Expand the first three levels of the search tree using A* and the heuristic h_1.

A* on 8-puzzle, h_1 heuristic

Only nodes in shaded area are expanded

Goal reached in node #13

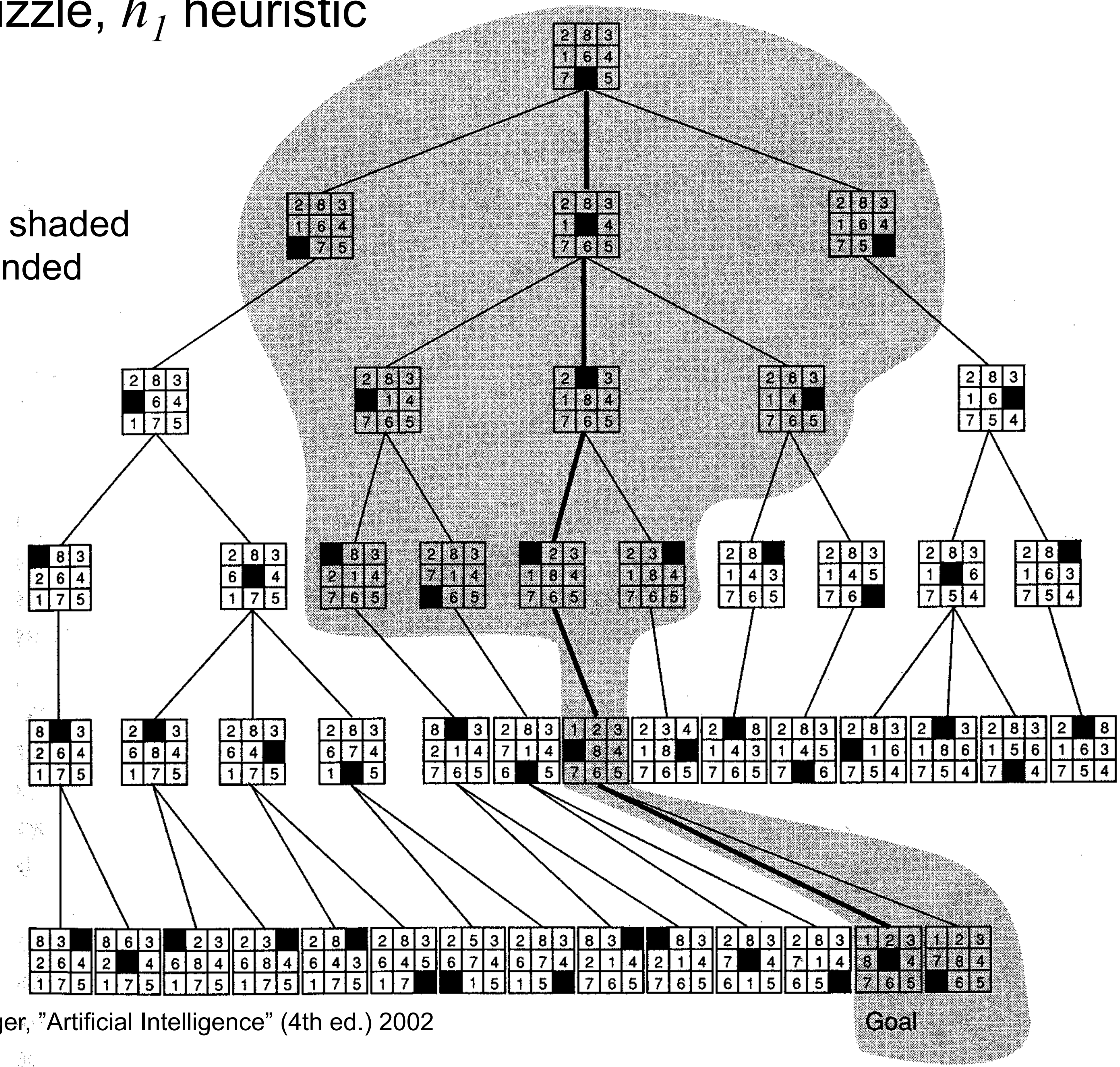


Image from G. F. Luger, ”Artificial Intelligence” (4th ed.) 2002

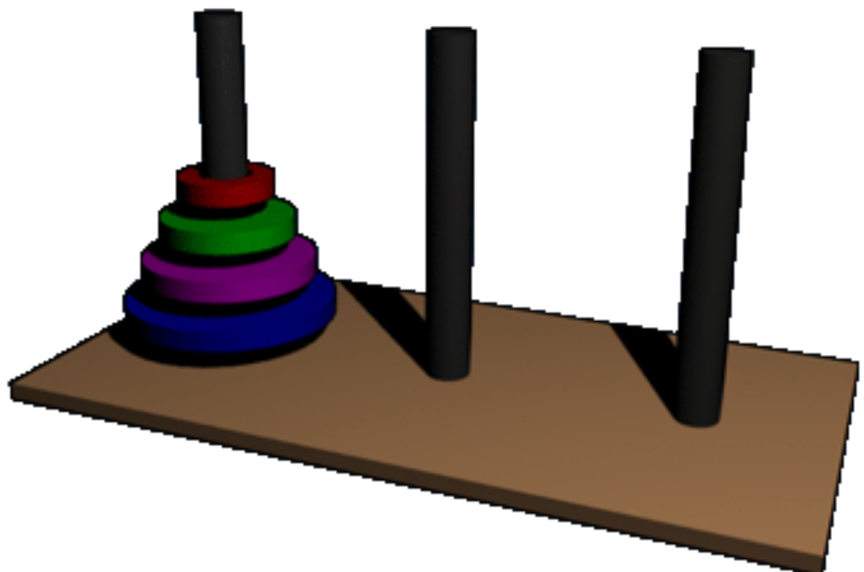
Domination

It is obvious from the definitions that
$h_1(n) \leq h_2(n)$.
We say that h_2 dominates h_1.

$$h_1(n) \leq h_2(n) \leq \text{true path cost to node } n$$

All nodes expanded with h_2 are also expanded with h_1 (but not vice versa). Thus, h_2 is better.

Exercise: Towers of Hanoi



Animation from http://www.eisbox.net/wp-uploads/archive/hanoi.gif

Three rods and N disks with holes in them (so that they can be placed on the rods).

The task is to move all disks from the leftmost rod on to the rightmost rod, without ever placing a larger disk on top of a smaller disk

Exercise: Towers of Hanoi

- Design a heuristic function for the Towers of Hanoi problem
  - Hint: Try to find an exact solution to the relaxed problem

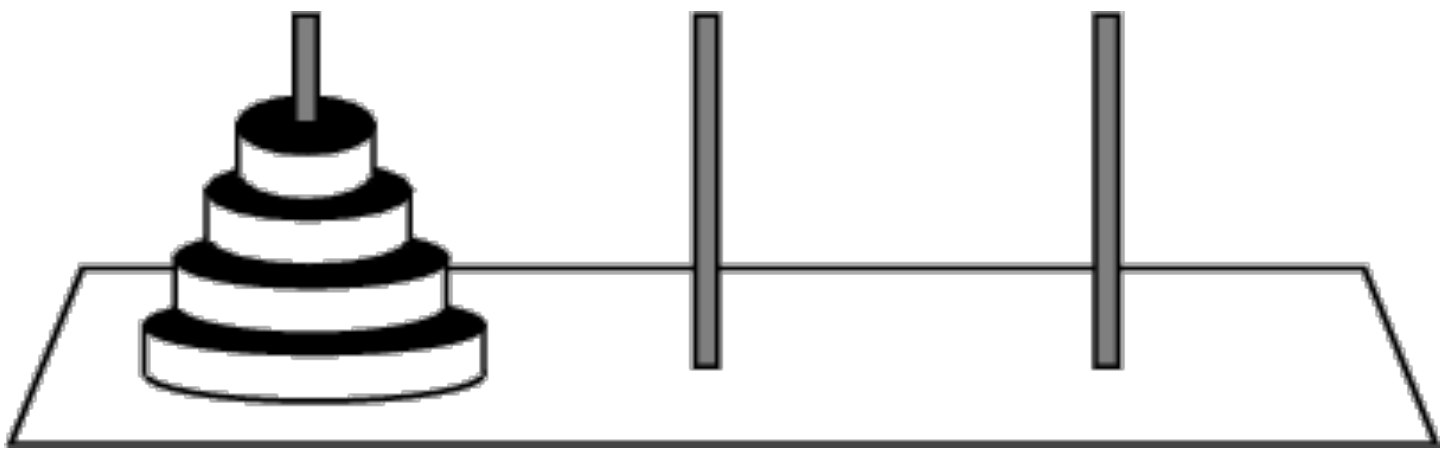


Image borrowed from http://mathworld.wolfram.com/TowersofHanoi.html

Exercise: Towers of Hanoi

Suggestions:

1. Number of disks on top of the largest disk when the largest disk is not in place

Exercise: Towers of Hanoi

Suggestions:

1. Number of disks on top of the largest disk when the largest disk is not in place
2. The number of disks that are not in place

Exercise: Towers of Hanoi

Suggestions:

1. Number of disks on top of the largest disk when the largest disk is not in place
2. The number of disks that are not in place
3. Sum of [2 × (the number of disks that are not in place)-1] over each non-goal peg, and add 2 × (the number of disks not in place) on the goal peg

Exercise: Towers of Hanoi

Suggestions:

1. Number of disks on top of the largest disk when the largest disk is not in place
2. The number of disks that are not in place
3. Sum of [2 × (the number of disks that are not in place)-1] over each non-goal peg, and add 2 × (the number of disks not in place) on the goal peg

Which one is best???

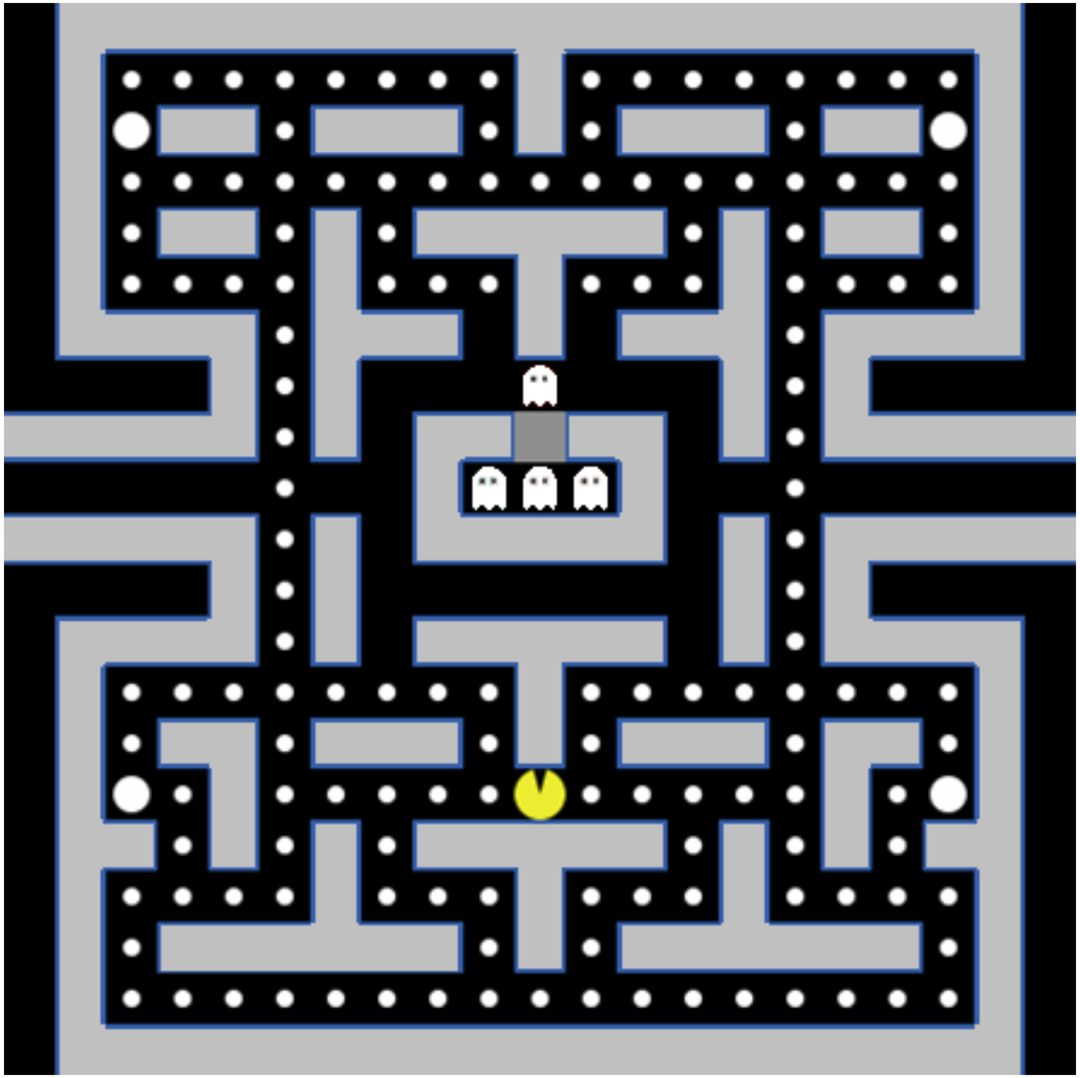
Exercise: Towers of Hanoi

Suggestions:

1. Number of disks on top of the largest disk when the largest disk is not in place
2. The number of disks that are not in place
3. Sum of [2 × (the number of disks that are not in place)-1] over each non-goal peg, and add 2 × (the number of disks not in place) on the goal peg

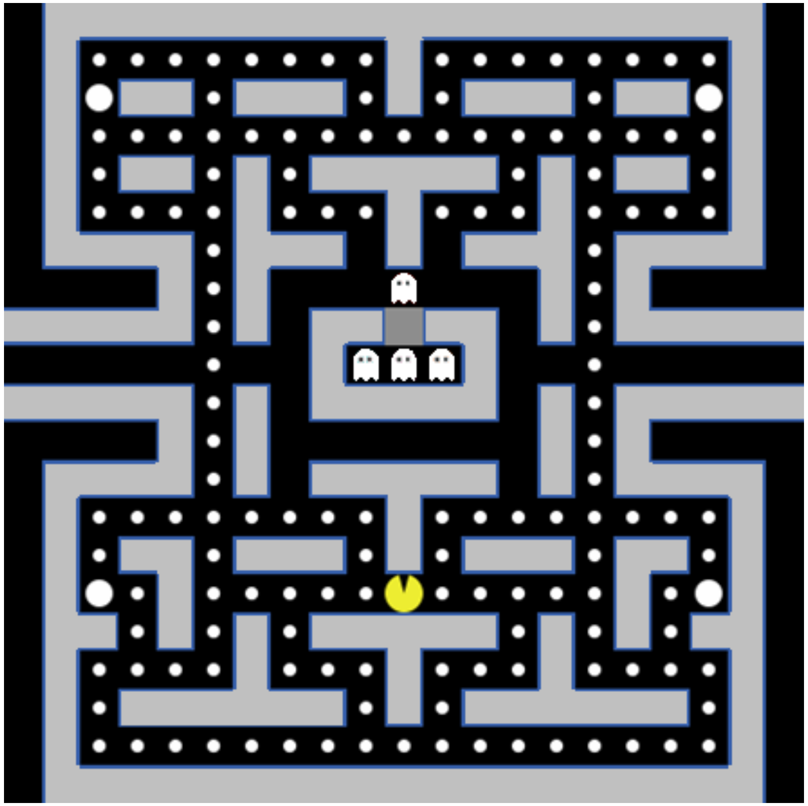
(1) < (2) < (3)

Example: Pac-Man



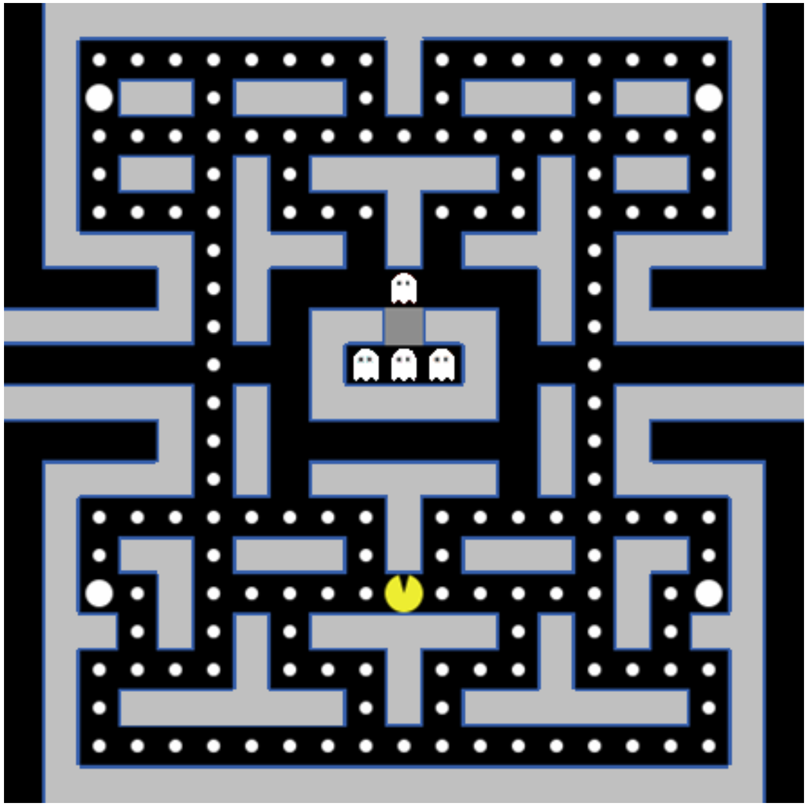
Example: Pac-Man

- Define the state space for the game
- What is the maximum branching factor for the game?
- For a game with N squares where Pac-Man can be (and that can be occupied by dots), what is the size of the state space?
- What is the goal test?
- Formulate a suitable heuristic for the game. Explain why it is a suitable heuristic.
- Will it make any difference which informed search method you use?



Example: Pac-Man

- Define the state space for the game
- What is the maximum branching factor for the game?
- For a game with N squares where Pac-Man can be (and that can be occupied by dots), what is the size of the state space?
- What is the goal test?
- Formulate a suitable heuristic for the game. Explain why it is a suitable heuristic.
- Will it make any difference which informed search method you use?



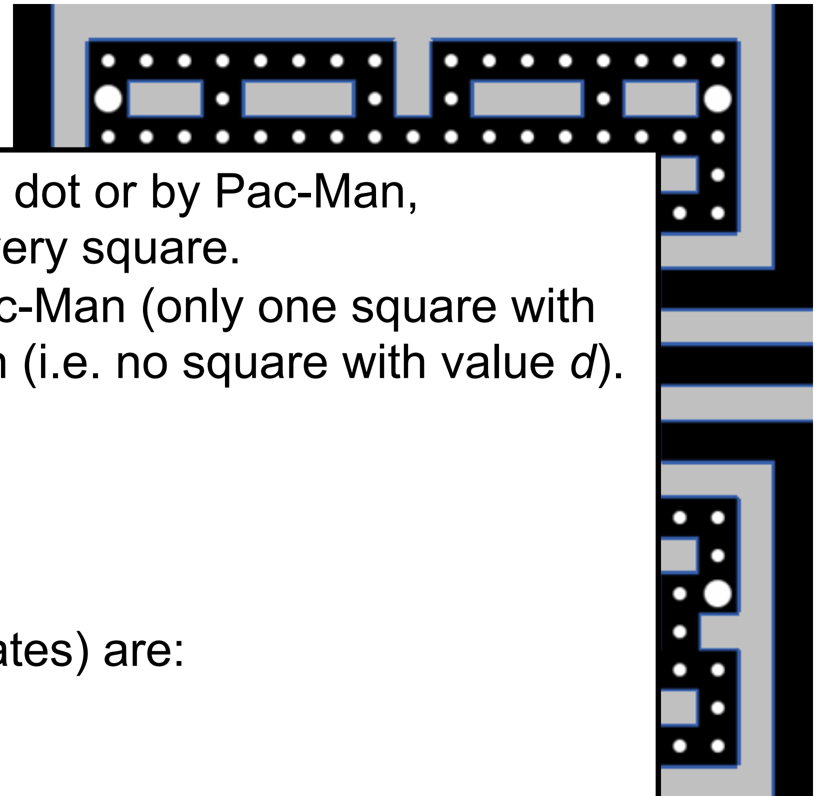
Example: Pac-Man

- Define the state space for the game

Each square can be empty, occupied by a dot or by Pac-Man,
i.e. we can have the values {*e, d, p*} for every square.
The constraint is that there is only one Pac-Man (only one square with value *p*). The task is to have all dots eaten (i.e. no square with value *d*).

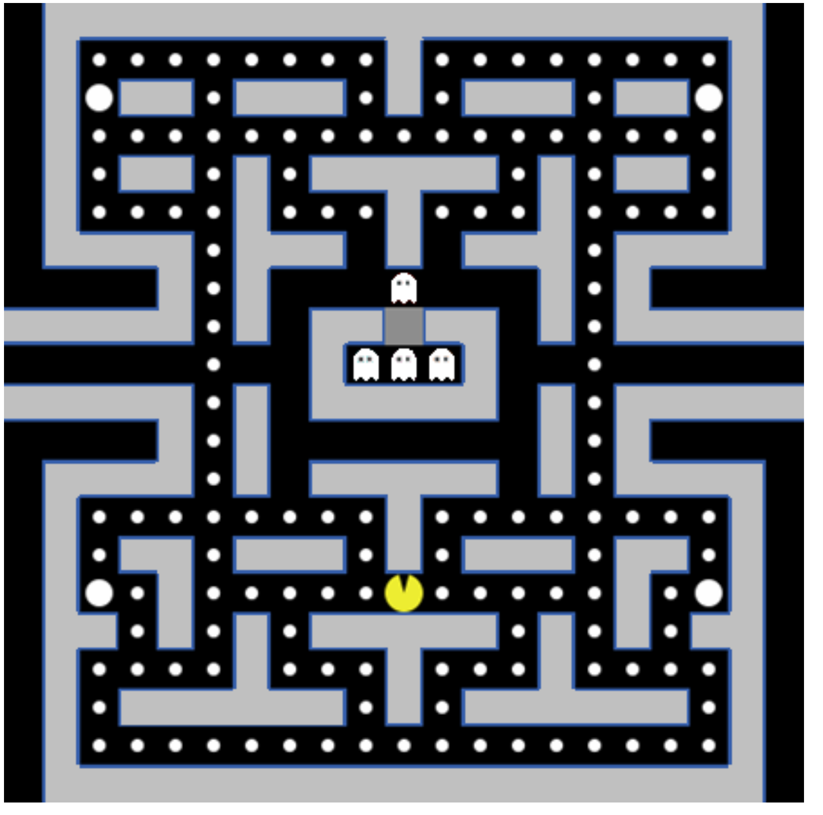
The initial state is then described as
{…, *d, d, d, d, p, d, d, d, d, …*}

The nodes that can be expanded (next states) are:
{…, *d, d, d, d, e, p, d, d, d, …*}
and
{…, *d, d, d, p, e, d, d, d, d, …*}



Example: Pac-Man

- Define the state space for the game
- What is the maximum branching factor for the game?
- For a game with N squares where Pac-Man can be (and that can be occupied by dots), what is the size of the state space?
- What is the goal test?
- Formulate a suitable heuristic for the game. Explain why it is a suitable heuristic.
- Will it make any difference which informed search method you use?

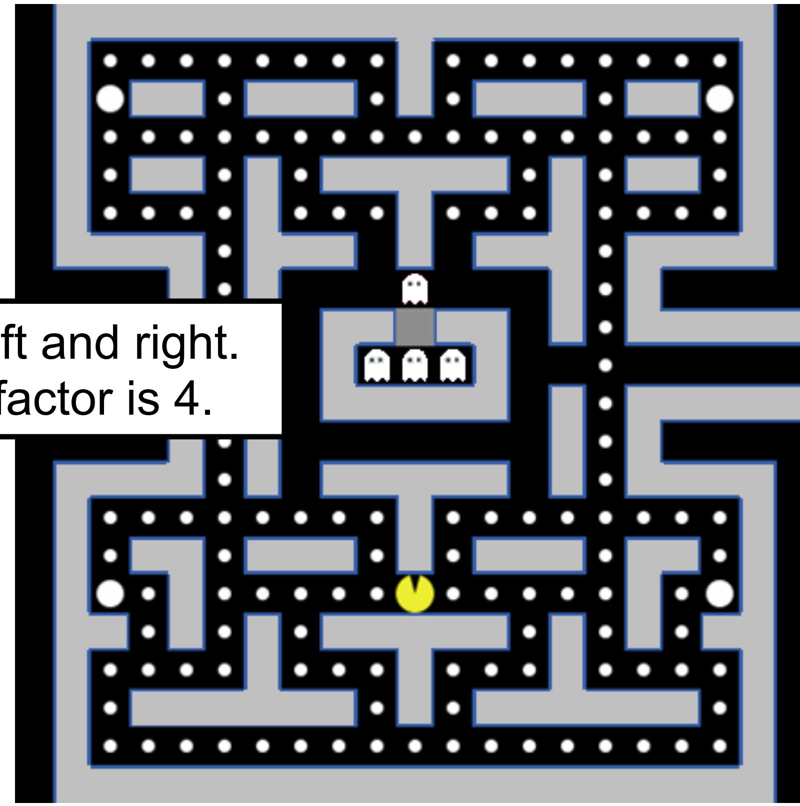


Example: Pac-Man

- Define the state space for the game
- What is the maximum branching factor for the game?
- For a game with N squares

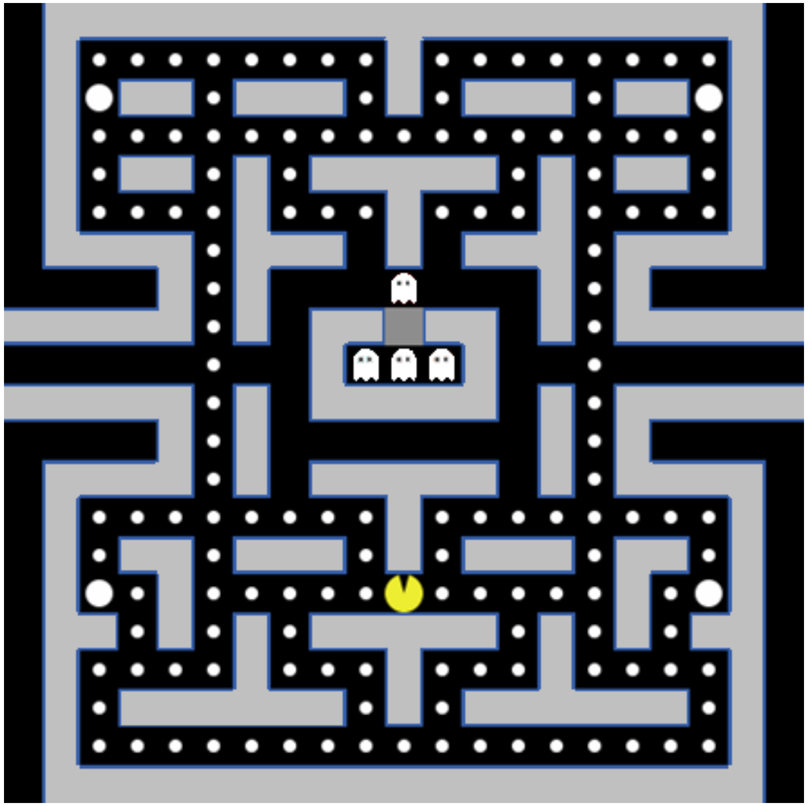
Pac-Man can move (at most) up, down, left and right.
This means that the maximum branching factor is 4.

  is the size of the state space?
- What is the goal test?
- Formulate a suitable heuristic for the game. Explain why it is a suitable heuristic.
- Will it make any difference which informed search method you use?



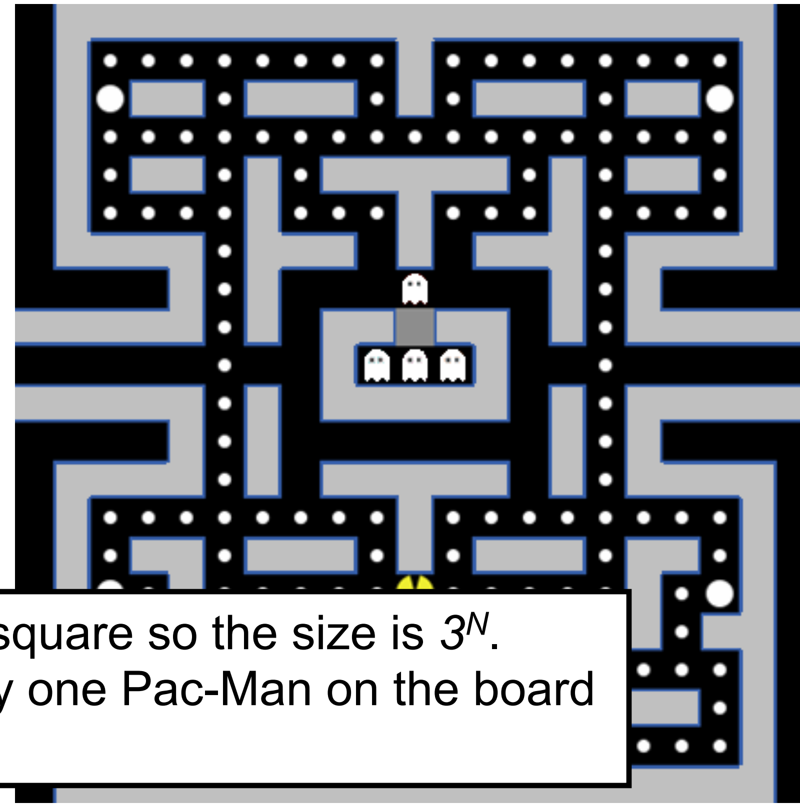
Example: Pac-Man

- Define the state space for the game
- What is the maximum branching factor for the game?
- For a game with N squares where Pac-Man can be (and that can be occupied by dots), what is the size of the state space?
- What is the goal test?
- Formulate a suitable heuristic for the game. Explain why it is a suitable heuristic.
- Will it make any difference which informed search method you use?



Example: Pac-Man

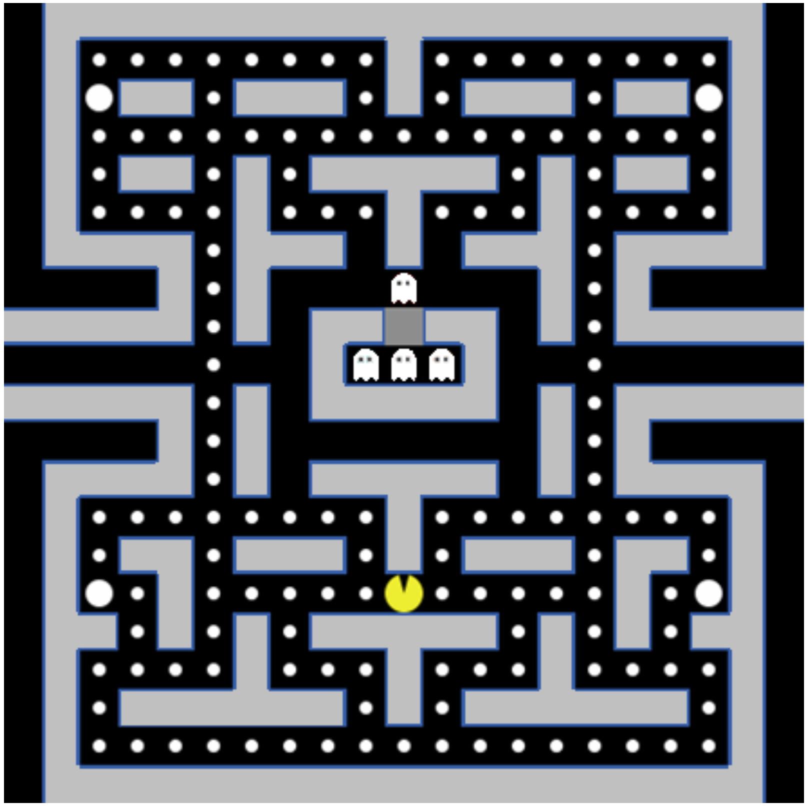
- Define the state space for the game
- What is the maximum branching factor for the game?
- For a game with N squares where Pac-Man can be (and that can be occupied by dots), what is the size of the state space?
- What is the goal test?
- Will it make any difference which informed search method you use?



There are three possible states for every square so the size is 3^N. (Actually quite a bit less since there is only one Pac-Man on the board but we are satisfied with this estimate.)

Example: Pac-Man

- Define the state space for the game
- What is the maximum branching factor for the game?
- For a game with N squares where Pac-Man can be (and that can be occupied by dots), what is the size of the state space?
- What is the goal test?
- Formulate a suitable heuristic for the game. Explain why it is a suitable heuristic.
- Will it make any difference which informed search method you use?

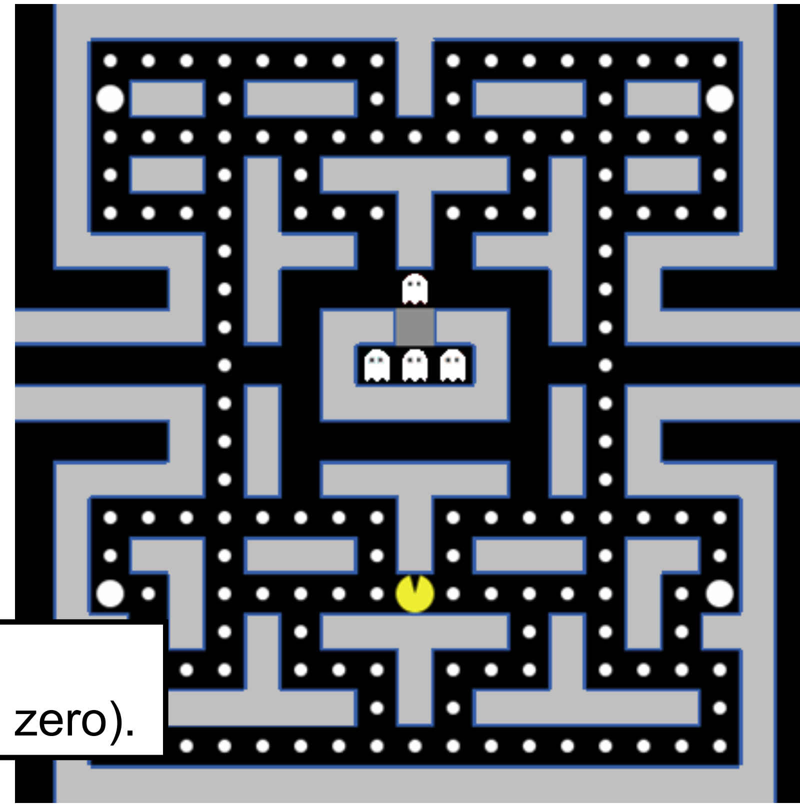


Example: Pac-Man

- Define the state space for the game
- What is the maximum branching factor for the game?
- For a game with N squares where Pac-Man can be (and that can be occupied by dots), what is the size of the state space?
- What is the goal test?
- Formulate a suitable heuristic for

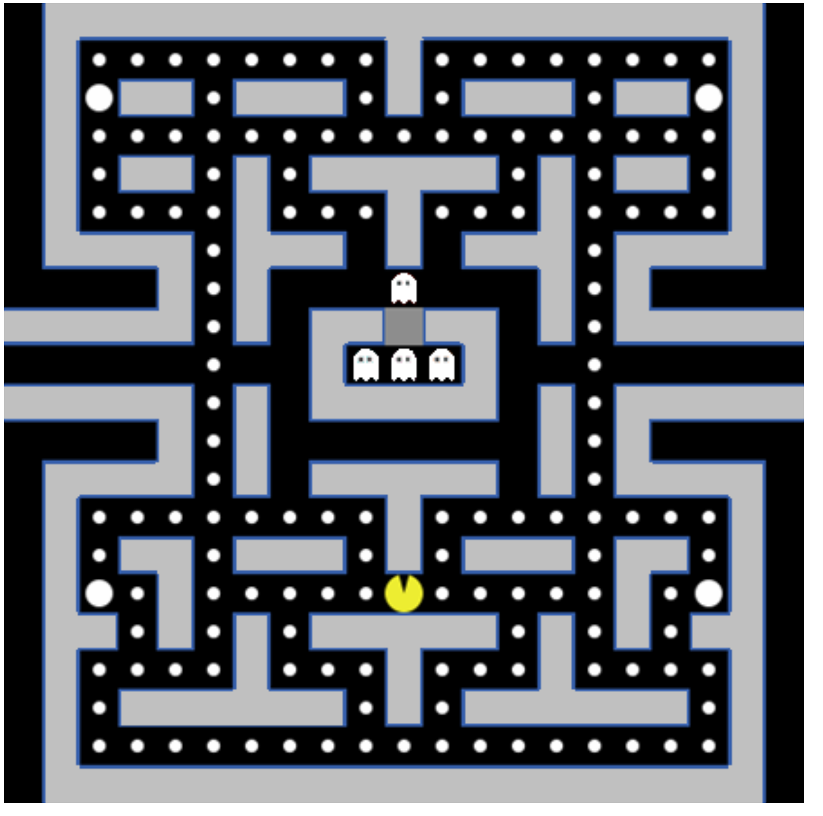
That there is no dot left in any square
(i.e. the number of squares with value *d* is zero).

- Will it make any difference which informed search method you use?



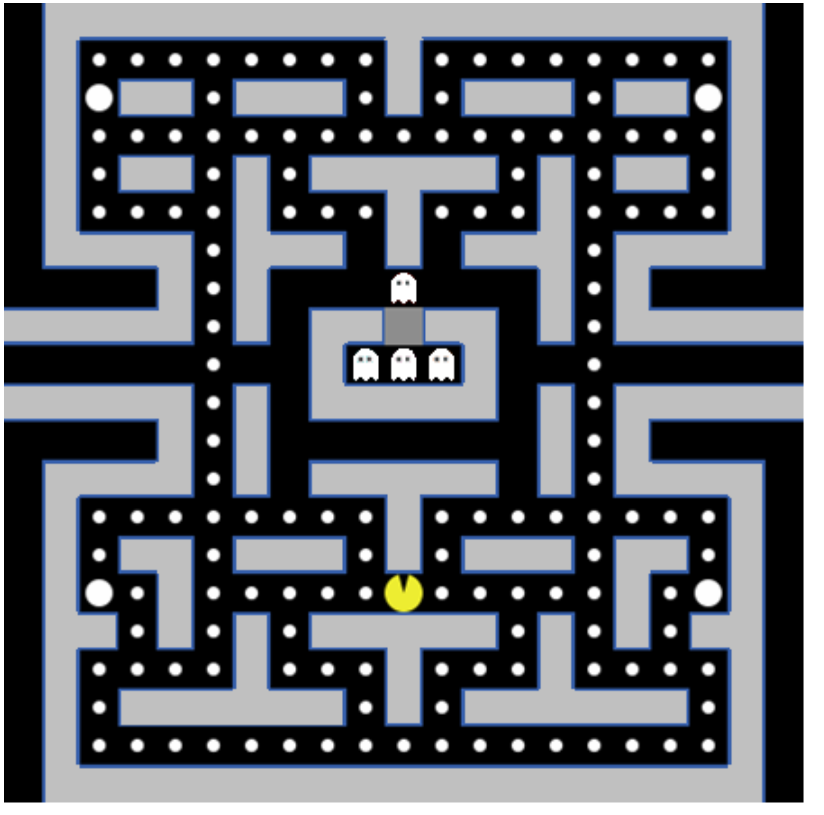
Example: Pac-Man

- Define the state space for the game
- What is the maximum branching factor for the game?
- For a game with N squares where Pac-Man can be (and that can be occupied by dots), what is the size of the state space?
- What is the goal test?
- Formulate a suitable heuristic for the game. Explain why it is a suitable heuristic.
- Will it make any difference which informed search method you use?



Example: Pac-Man

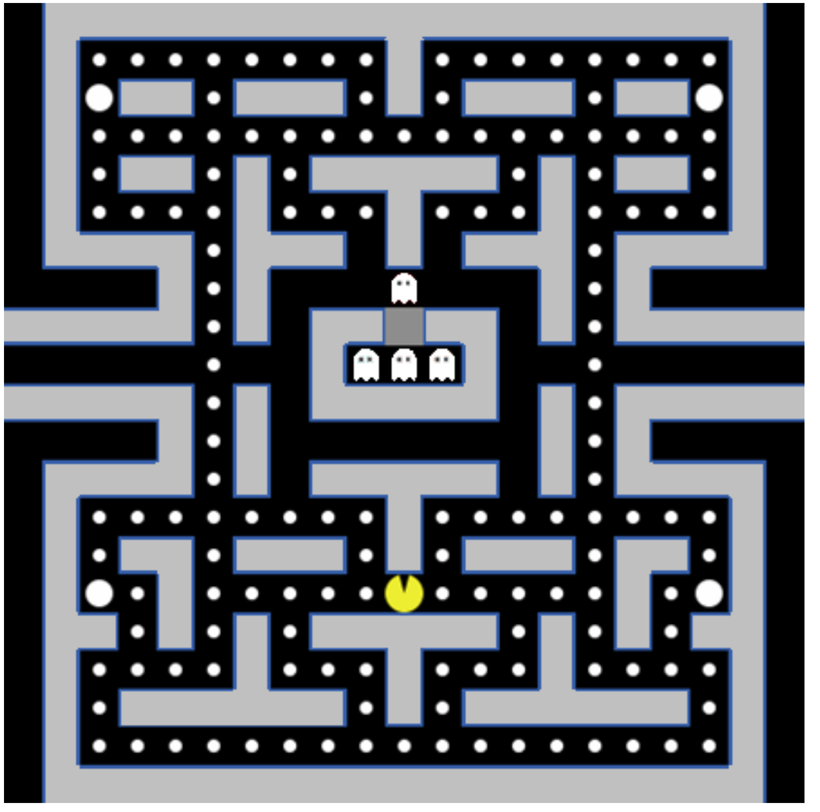
- Define the state space for the game
- What is the maximum branching factor for the game?
- For a game with N squares where Pac-Man can be (and that can be occupied by dots), what is the size of the state space?
- What is the goal test?
- Formulate a suitable heuristic for the game. Explain why it is a suitable heuristic.
- Will it make any difference which



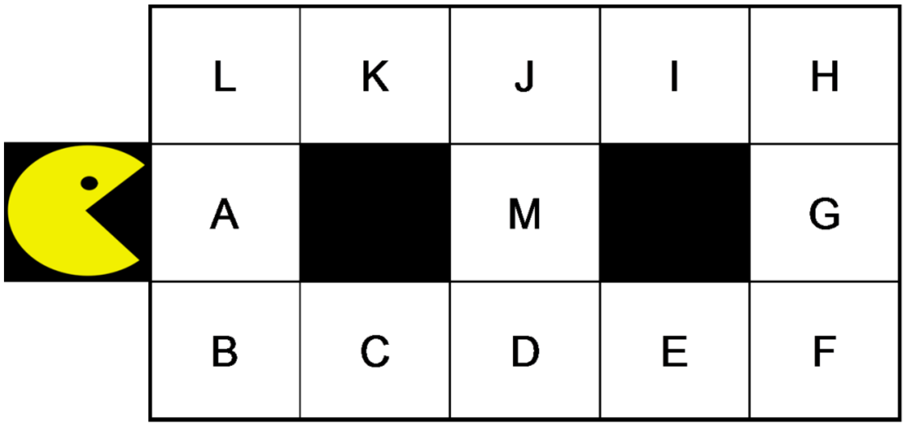
The number of squares with value *d*, this is a minimum number of moves that Pac-Man must do to empty all the squares.

Example: Pac-Man

- Define the state space for the game
- What is the maximum branching factor for the game?
- For a game with N squares where Pac-Man can be (and that can be occupied by dots), what is the size of the state space?
- What is the goal test?
- Formulate a suitable heuristic for the game. Explain why it is a suitable heuristic.
- Will it make any difference which informed search method you use?



Example: Pac-Man



- Will it make any difference which informed search method you use?

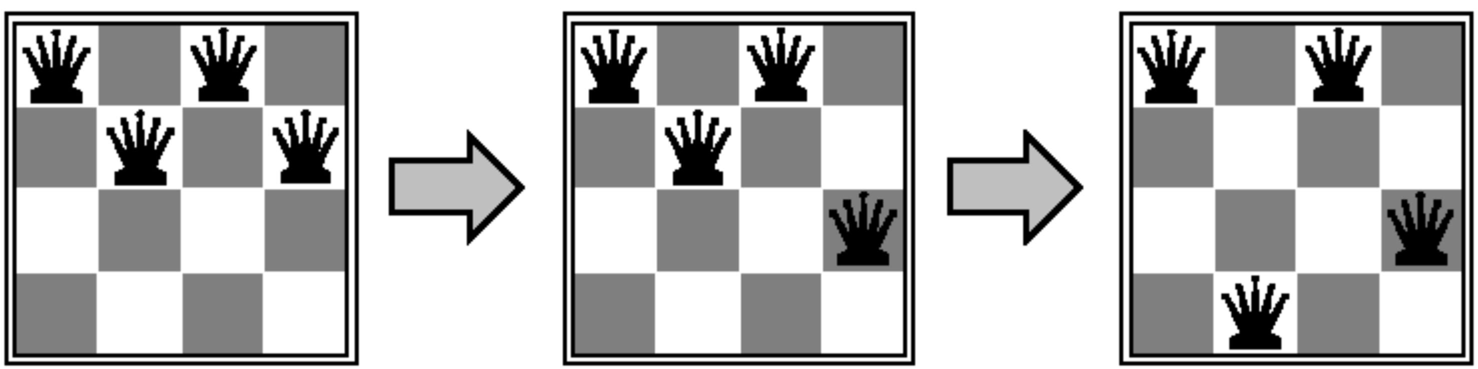
- Pac-Man is about to eat the dots in the squares A through M, the question is just what the optimal order is (with minimum number of steps).
- An optimal order is (e.g.) A-B-C-D-E-F-G-H-I-J-M-J-K-L = 14 moves. A suboptimal order is (e.g.) A-B-C-D-E-F-G-H-I-J-K-L-K-J-M = 15 moves.
- The suboptimality of the latter order is not discovered until the 14th move.
- A* is better than GBFS (we know this from theory) but there may be no difference between their performance.

Local search

Chapter 4 in AIMA 3rd ed

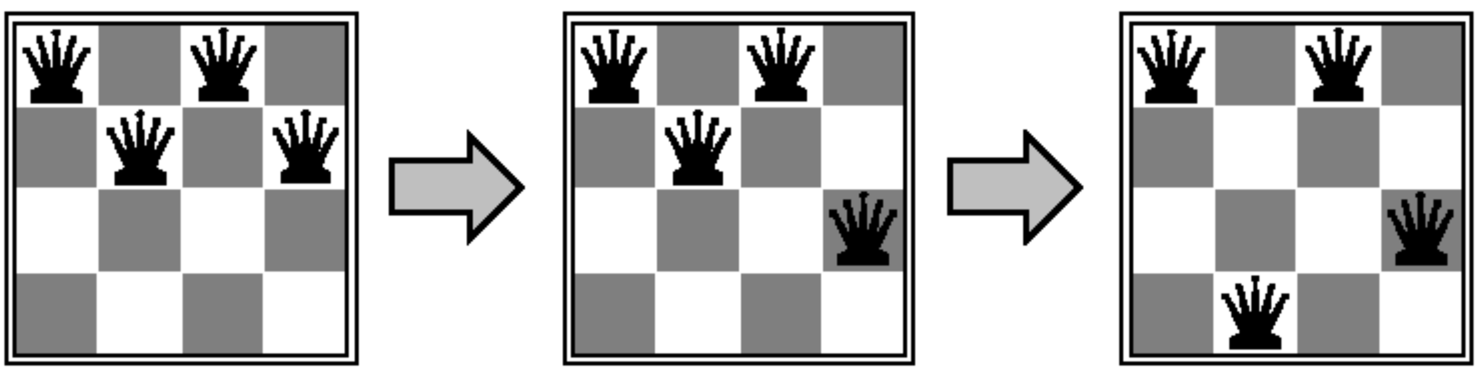
Example: N-queens

From initial state (in N × N chessboard), try to move to other configurations such that the number of conflicts is reduced.



Hill-climbing

- Current node = n_i.
- Grab a neighbor node n_{i+1} and move there if it improves things, i.e. if $\Delta f = f(n_i) - f(n_{i+1}) > 0$



Local beam search

- Start with k random states
- Expand all k states and test their children states.
- Keep the k best children states
- Repeat until goal state is found or performance is satisfactory

Genetic algorithms

- Start with k random states
- Selective breeding by mating the best states (with mutation)