CSC242: Introduction to Artificial Intelligence

Lecture 1.1

Please put away all electronic devices

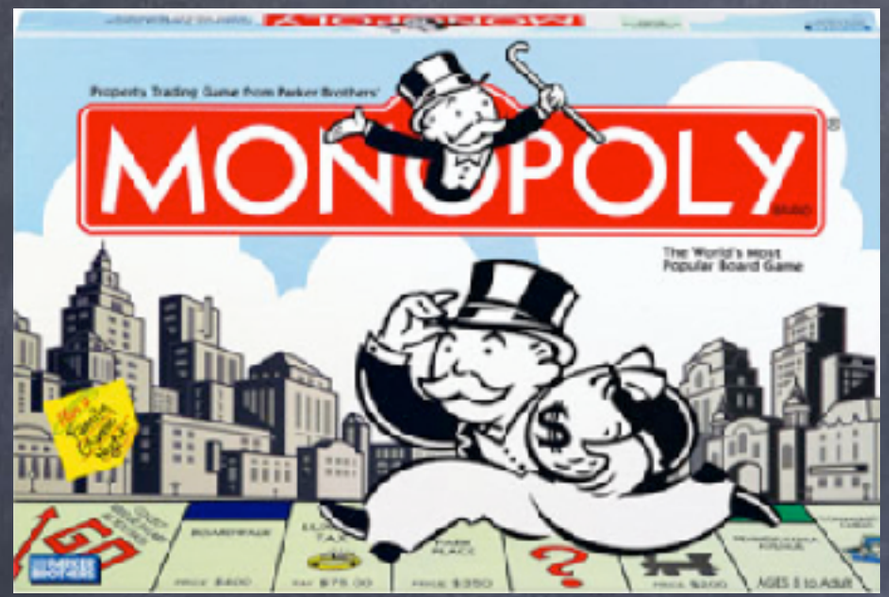
Announcements

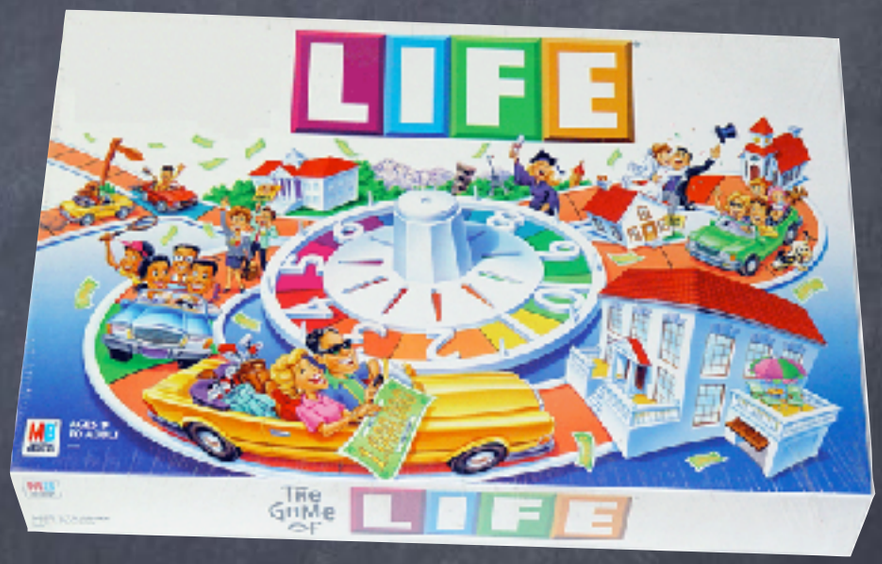
- Sorry for running late last class!
  - Will not happen again
- Study Sessions:
  - Mon & Wed 745PM-9PM, room TBA
- Project 1 will be posted after class

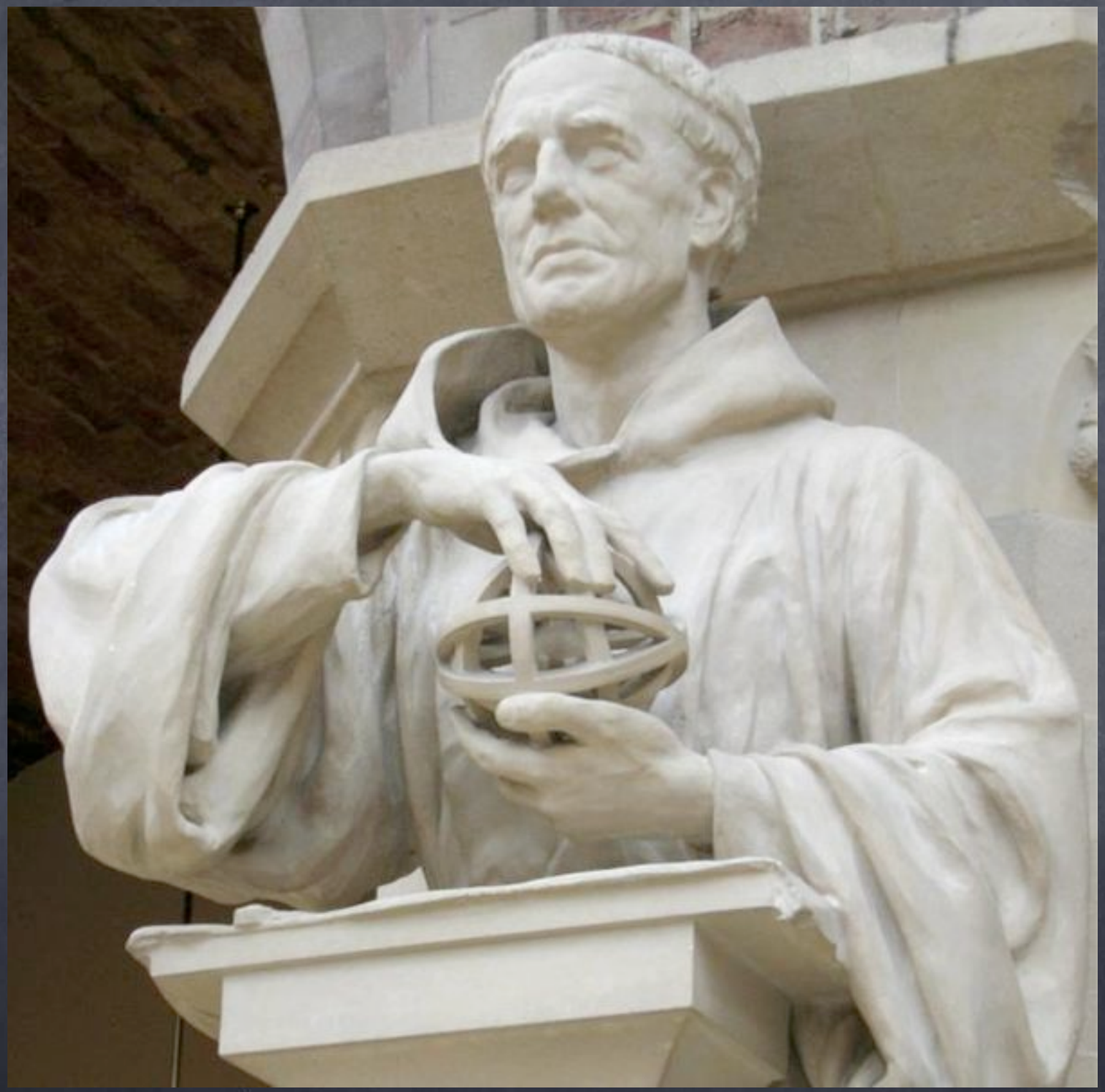
Problem Solving

Toy Problems

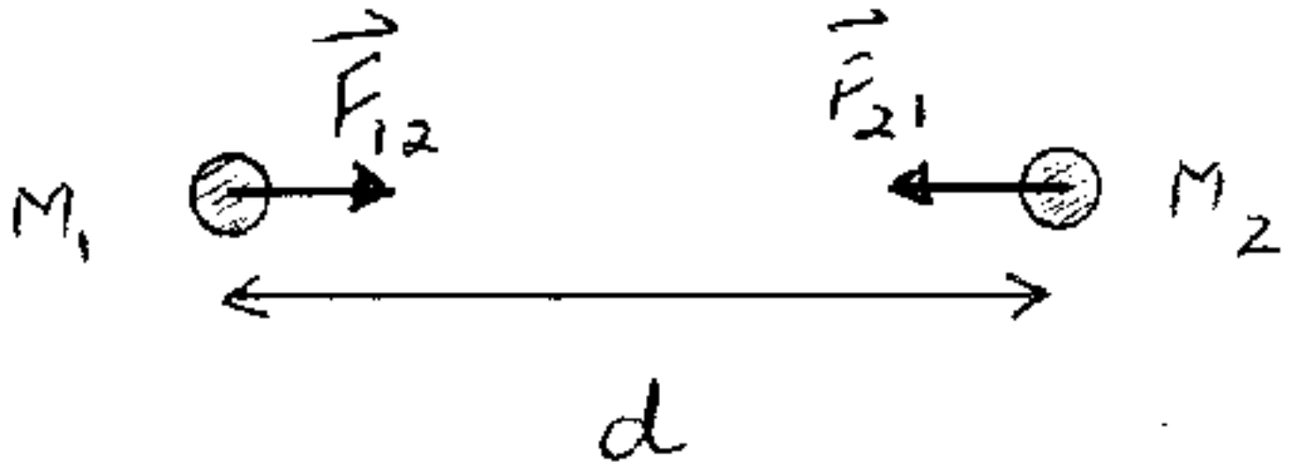
Toy Problems







Roger Bacon (c. 1214–1294)



$$|\vec{F}_{12}| = |\vec{F}_{21}| = \frac{G M_1 M_2}{d^2}$$

Abstraction

Simplification and generalization of real-world phenomena

Abstraction

We "abstract" away the details whose effect on the solution to a problem is minimal or nonexistent, thereby creating a model that lets us deal with the essence of the problem.

– Aho & Ullman,
Foundations of Computer Science

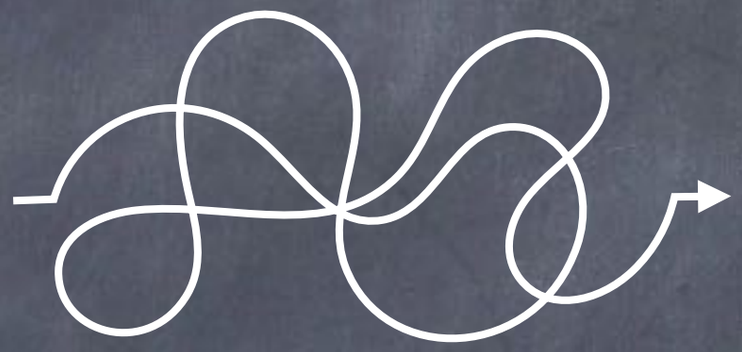
Abstraction

Fundamentally, Computer Science is a science of abstraction — creating the right model for thinking about a problem and devising the appropriate mechanizable techniques to solve it.

– Aho & Ullman,
Foundations of Computer Science

Real World

Complex Questions



Answers

Abstraction

Application

Model

Simple Questions

Answers

Our First Problem

RETEAUA DRUMURILOR NATIONALE DIN ROMANIA

M.T.

Legendă

Drumuri naționale europene	6.167 Km
din care:	
Autostrăzi	211 Km
Drumuri naționale principale	4.388 Km
Drumuri naționale secundare	5.442 Km
Total	15.997 Km
CF	

C.N.A.D.N.R.

S.C. IPTANA S.A.


Miskolc
Eger
Nyíregyháza
Satu Mare
Sighetu Marmației
Rădăuți
Baia Mare
Suceava
Debrecen
Balti
Moldova
Szolnok
Zalău
Vatra Dornei
Iași
Oradea
Bistrița
Chisinau
Piatra Neamț
Tiraspol
Cluj-Napoca
Gyula
Turda
Târgu Mureș
Vaslui
Bacău
Arad
Onești
Bârlad
Sighișoara
Alba Iulia
Deva
Romania
Timișoara
Focșani
Zrenjanin
Зрењанин
Reșița
Galați
Brăila
Izmail
Ізмаїл
Tulcea
Rezervația Biosferei Delta Dunării
Sinaia
Râmnicu Vâlcea
Buzău
Târgu Jiu
Belgrade
Београд
Ploiești
Pitești
Drobeta-Turnu Severin
Slatina
Bucharest
Craiova
Călărești
Constanța
Eforie Nord
Vidin
Видин
Serbia
Mangalia
Ruse
Русе
Kruševac
Крушевац
Dobrich
Добрич
Niš
Pleven

RETEAUA DRUMURILOR NATIONALE DIN ROMANIA

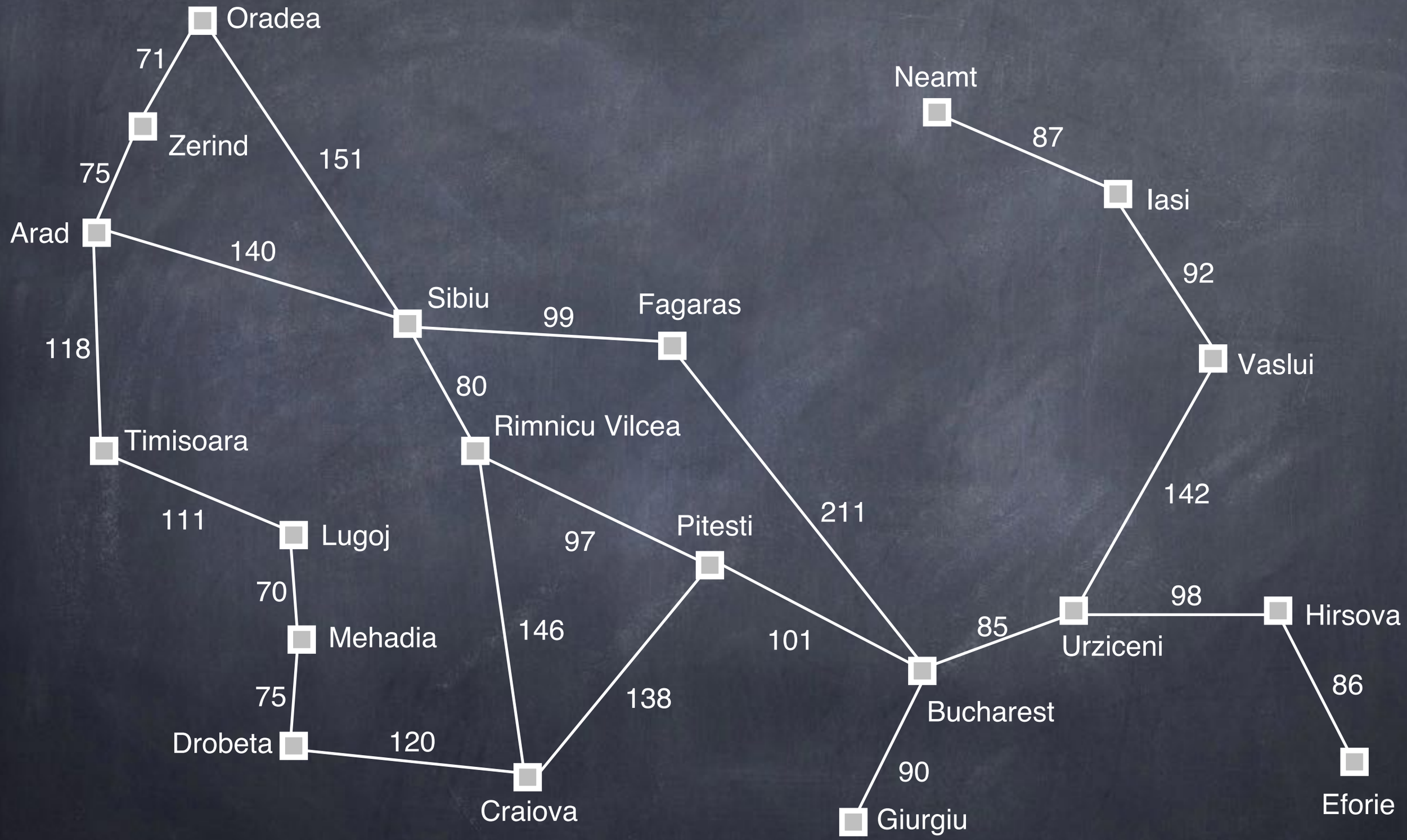
M.T.

Legendă

Drumuri naționale europene	6.167 Km
din care:	
Autostrăzi	211 Km
Drumuri naționale principale	4.388 Km
Drumuri naționale secundare	5.442 Km
Total	15.997 Km
CF	

C.N.A.D.N.R.

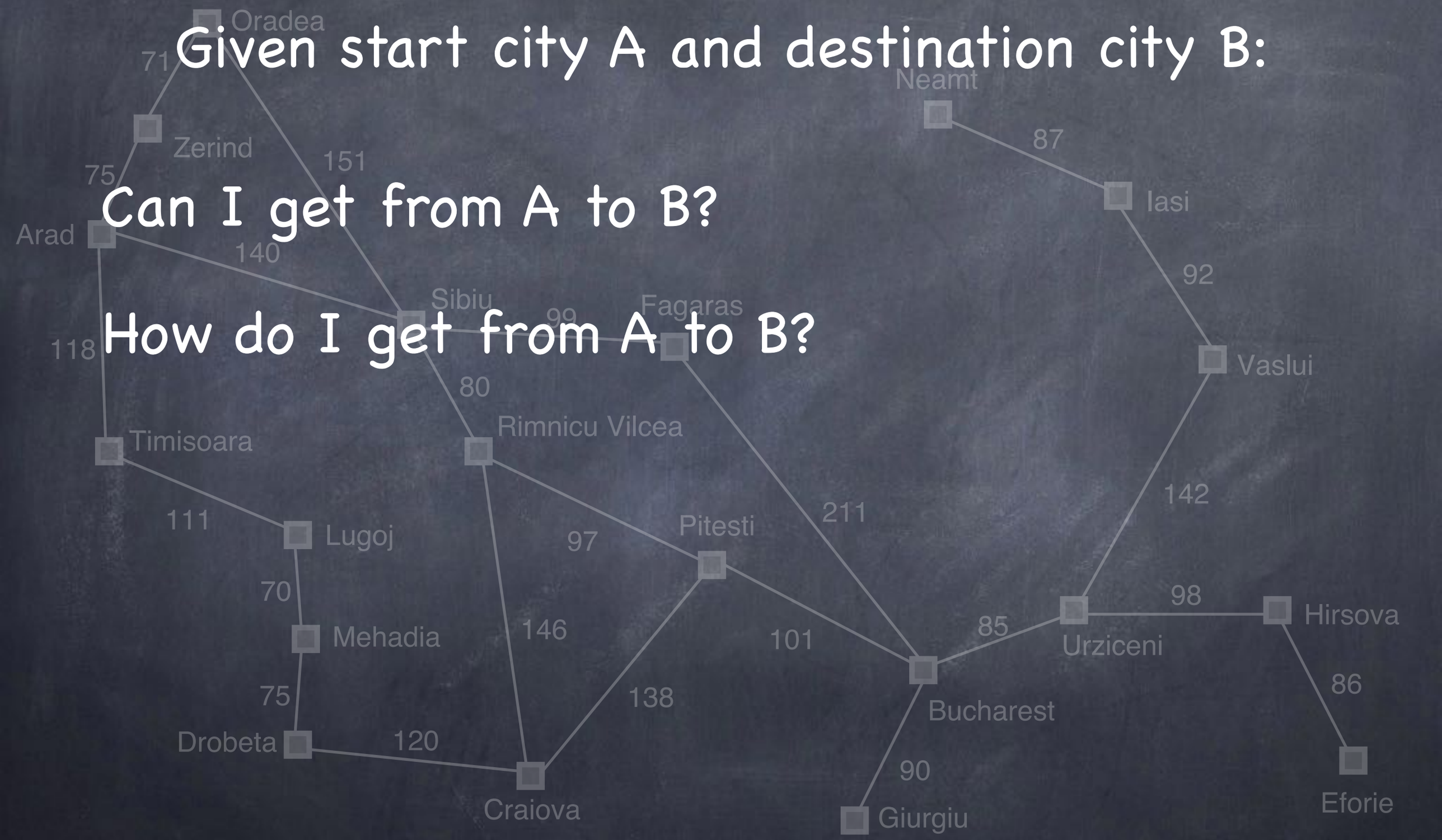
S.C. IPTANA S.A.


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

Given start city A and destination city B:

Can I get from A to B?

How do I get from A to B?



Given start city A and destination city B:

Can I get from A to B?

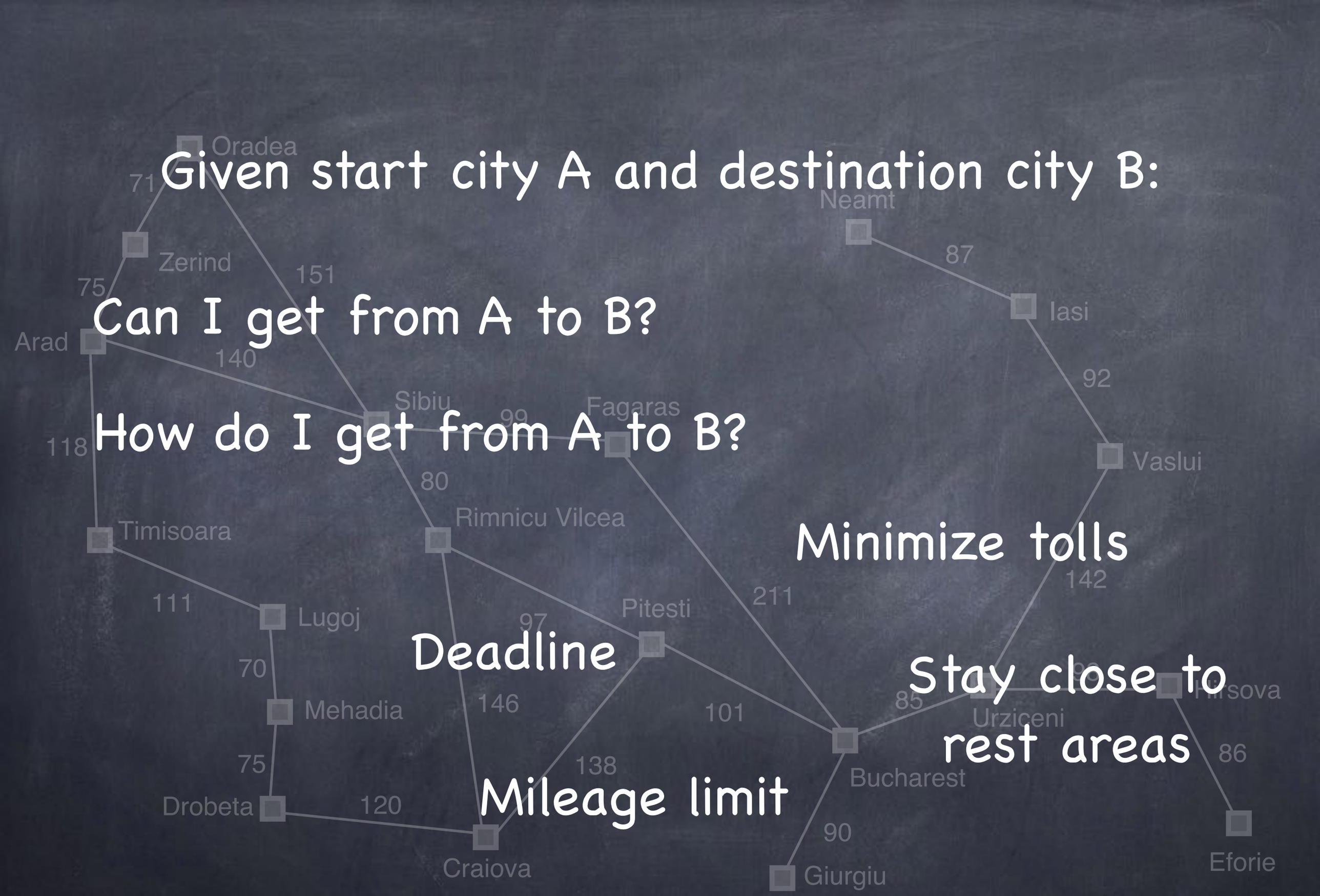
How do I get from A to B?

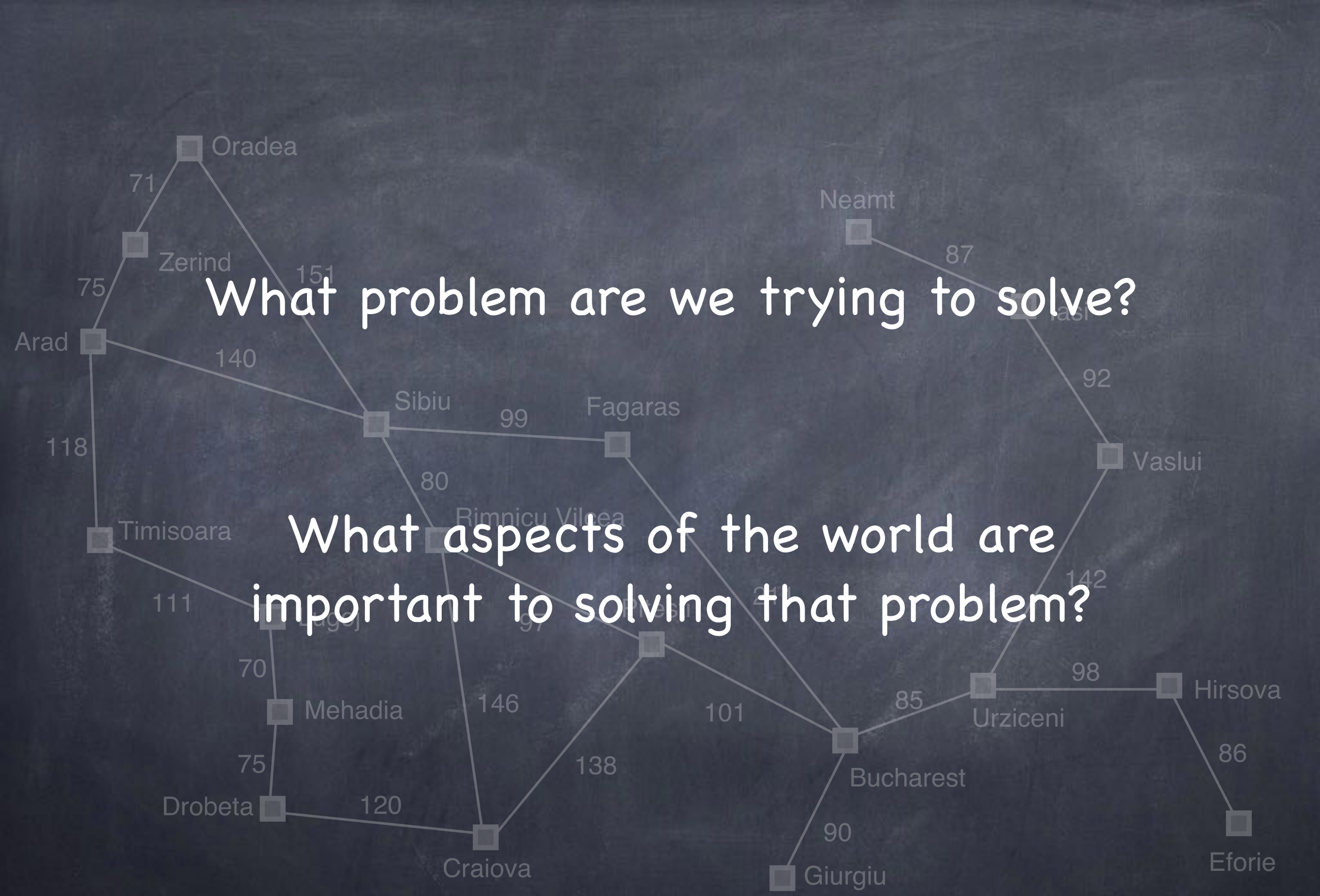
Minimize tolls

Deadline

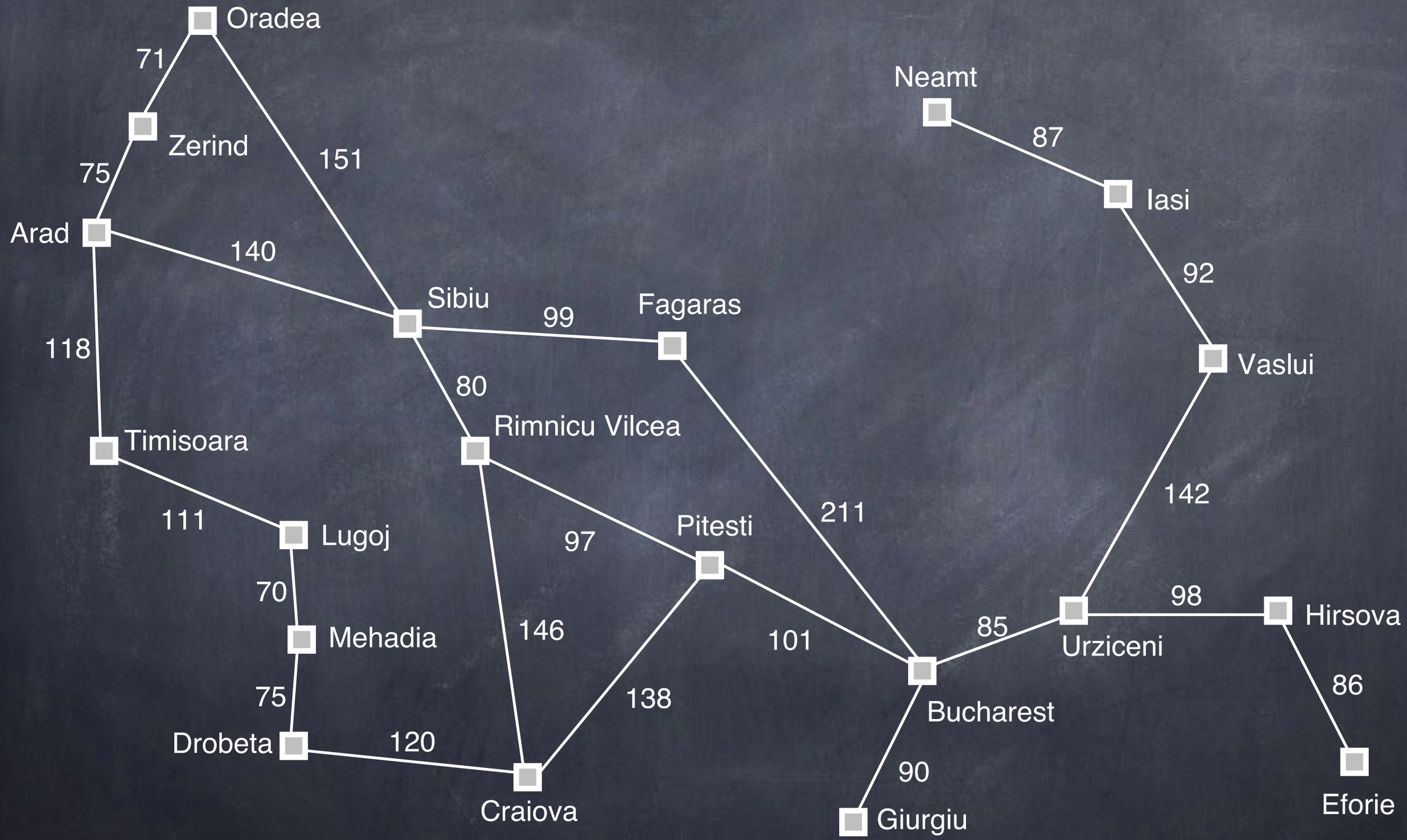
Stay close to rest areas

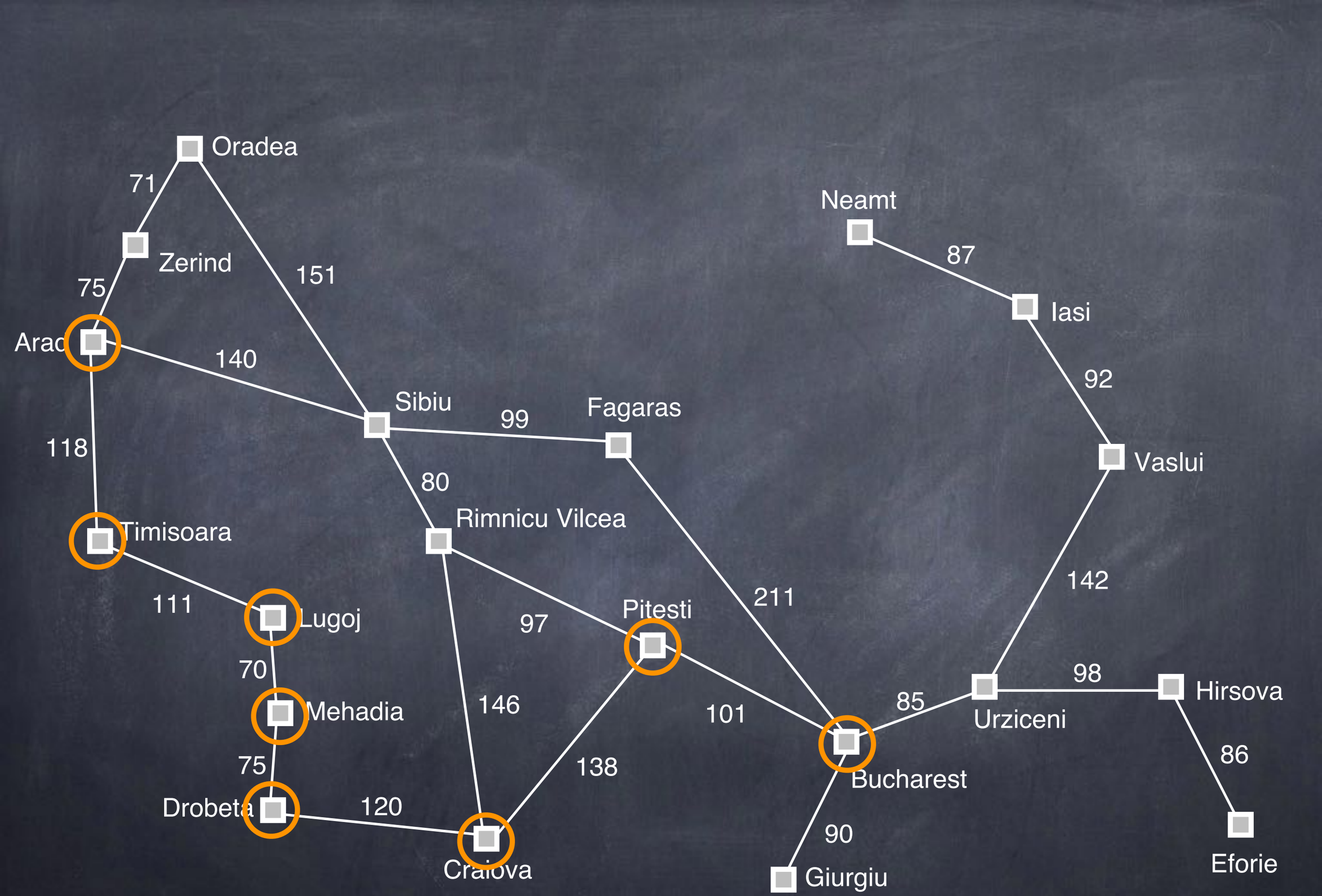
Mileage limit




Oradea
71
Neamt
Zerind
87
151
75
What problem are we trying to solve?
Arad
140
92
Sibiu
Fagaras
99
118
Vaslui
80
What aspects of the world are important to solving that problem?
Timisoara
Rimnicu Vilcea
111
142
97
98
70
Hirsova
Mehadia
146
85
Urziceni
101
75
86
138
Bucharest
Drobeta
120
90
Craiova
Giurgiu
Eforie

Problem Solving by Computers


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


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

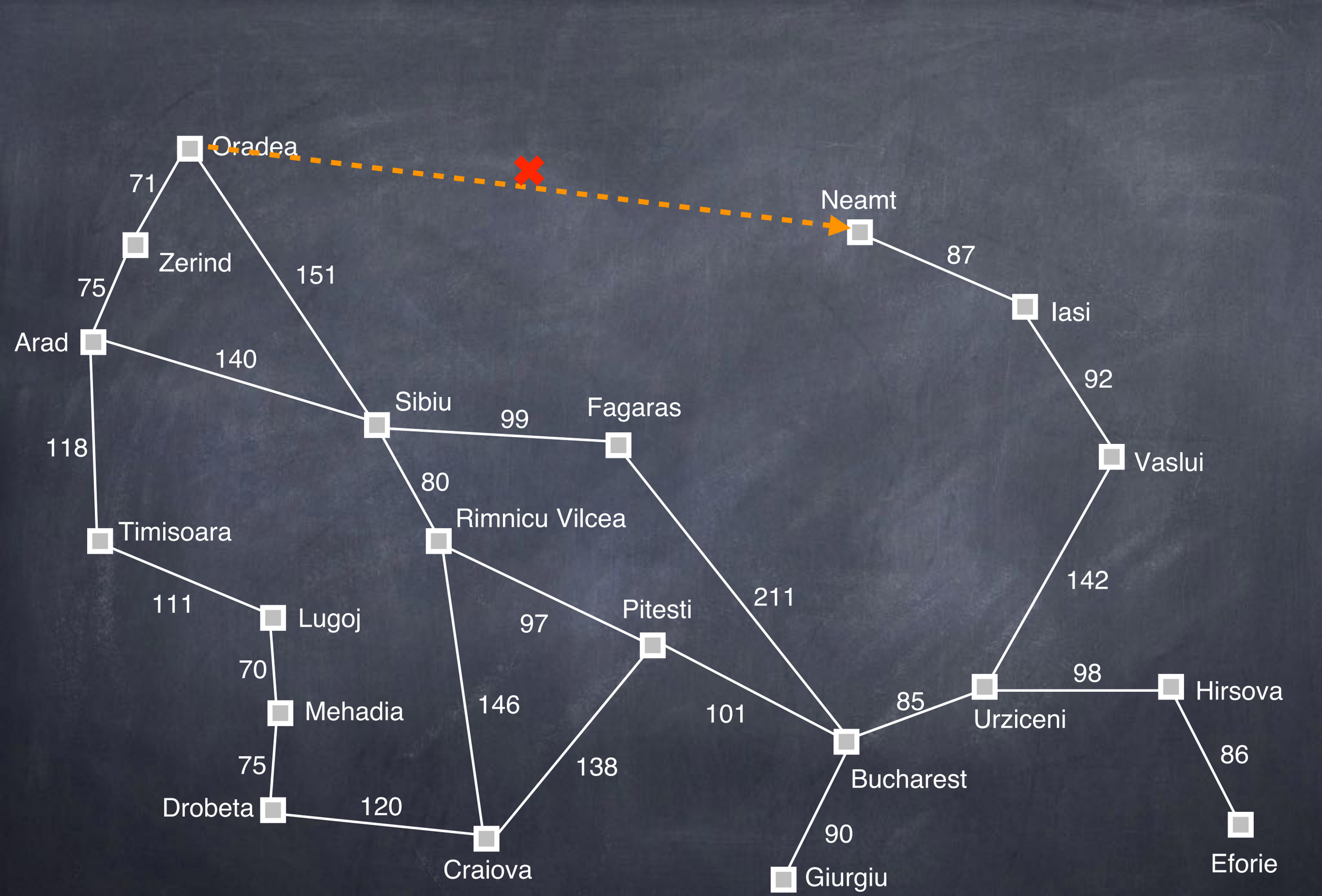
State (of the World)

- A representation of those aspects of the world that matter to solving the problem


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

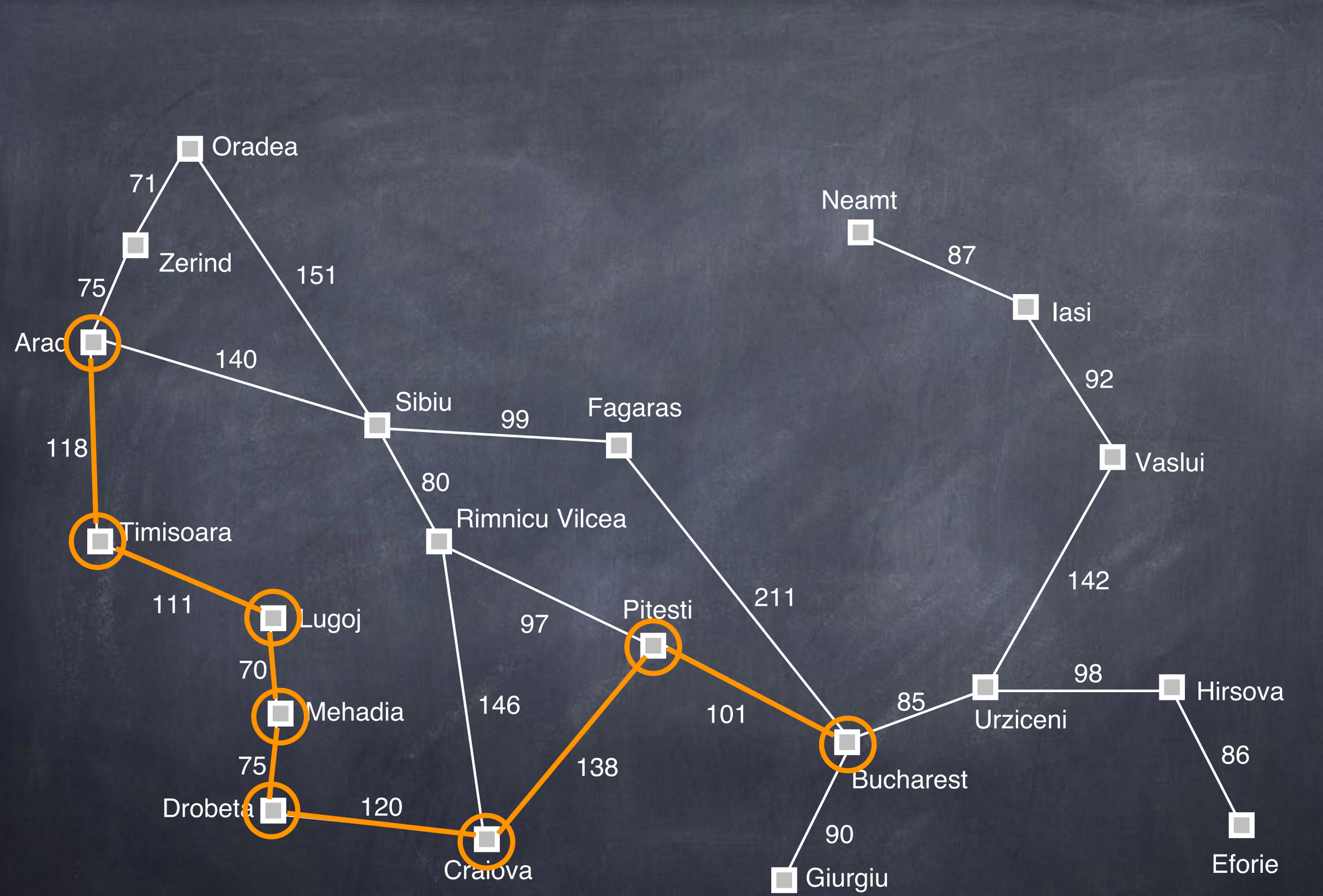
Actions

- Change the state of the world
  - Cause a transition from one state to another


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

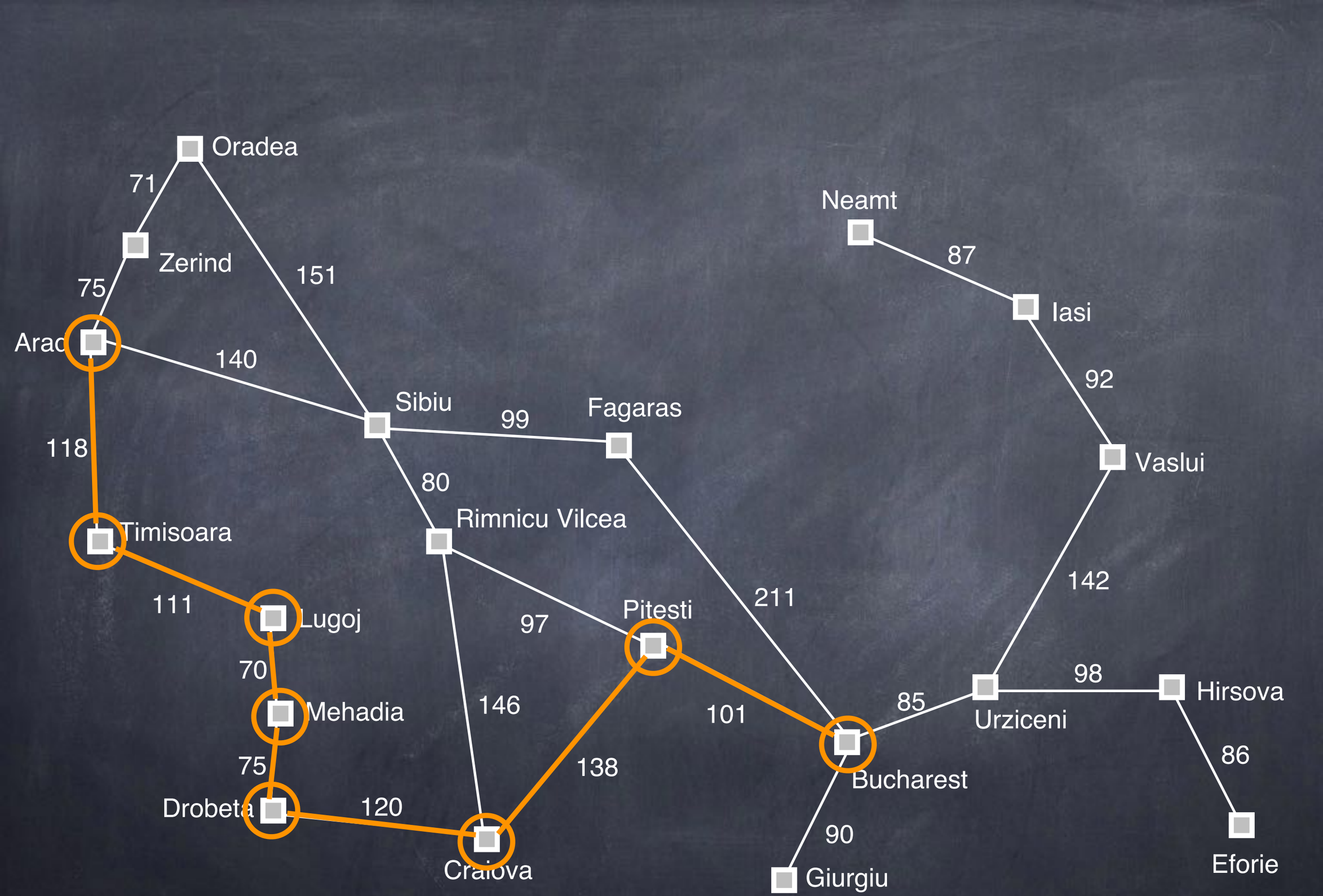
Actions

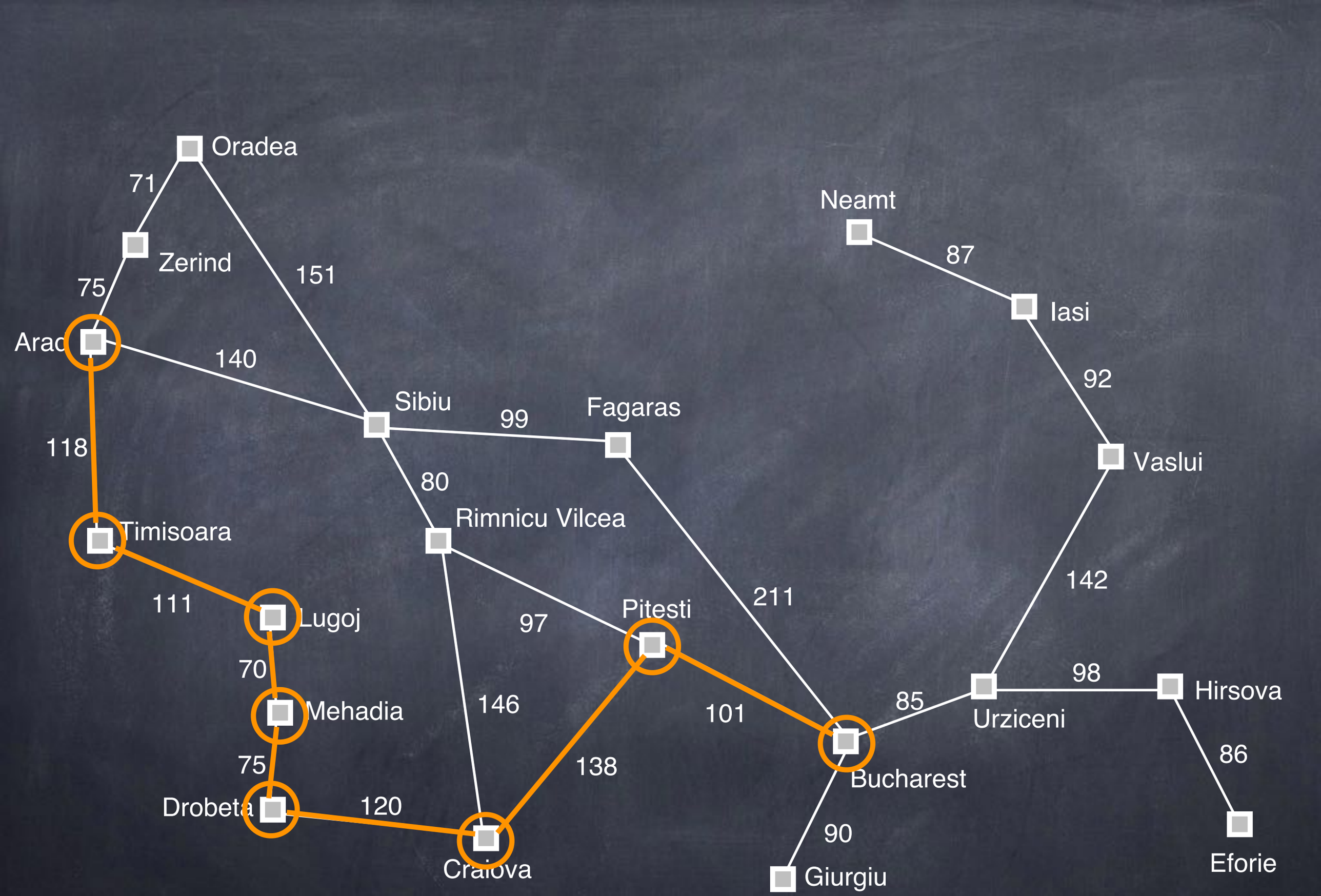
- Change the state of the world
  - Cause a transition from one state to another
  - Not all actions are applicable in every state


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

Solutions

- Solutions are sequences of actions that lead from the initial state to a goal state


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



118+111+70+75+120+138+101=733

Modeling Problems (and their Solutions)

- Model the world as a set of states
- Actions change the state of the world
- Cost of solution (path cost)
  - Cost of individual actions (step cost)
- Solutions are sequences of actions that lead from the initial state to a goal state

Problem (Domain): $\langle S,\ A,\ \textsc{Actions},\ \textsc{Result},\ \textsc{Cost} \rangle$

$\textsc{Actions} : s \in S \rightarrow$

$\{\ a \in A : a \text{ can be executed (is applicable) in } s\ \}$

$\textsc{Result} : s \in S\ ,\ a \in A \rightarrow$

$s' \in S$ s.t. s' is the result of performing a in s

$\textsc{Cost}$: Assigns a cost to each path/step $c(s, a, s')$

Problem (Domain): $\langle S,\ A,\ \textsc{Actions},\ \textsc{Result},\ \textsc{Cost} \rangle$

Transition Model

$\textsc{Actions} : s \in S \rightarrow$

$\{\ a \in A : a \text{ can be executed (is applicable) in } s\ \}$

$\textsc{Result} : s \in S\ ,\ a \in A \rightarrow$

$s' \in S$ s.t. s' is the result of performing a in s

$\textsc{Cost}$: Assigns a cost to each path/step $c(s, a, s')$

Problem (Domain): $\langle S,\ A,\ \textsc{Actions},\ \textsc{Result},\ \textsc{Cost} \rangle$

Transition Model

$\textsc{Actions} : s \in S \rightarrow$

$\{\ a \in A : a \text{ can be executed (is applicable) in } s\ \}$

$\textsc{Result} : s \in S\ ,\ a \in A \rightarrow$

$s' \in S$ s.t. s' is the result of performing a in s

$\textsc{Cost}$: Assigns a cost to each path/step $c(s, a, s')$

Problem (Instance): $\langle I \in S,\ G \subseteq S \rangle$

Problem (Domain): $\langle S,\ A,\ \textsc{Actions},\ \textsc{Result},\ \textsc{Cost} \rangle$

$\textsc{Actions} : s \in S \rightarrow$

$\{\ a \in A : a \text{ can be executed (is applicable) in } s\ \}$

$\textsc{Result} : s \in S\ ,\ a \in A \rightarrow$

$s' \in S$ s.t. s' is the result of performing a in s

$\textsc{Cost}$: Assigns a cost to each path/step $c(s, a, s')$

Problem (Instance): $\langle I \in S,\ G \subseteq S \rangle$

Solution: $\langle a_1,\ a_2,\ \ldots,\ a_n \rangle \in A^n$ s.t.

$\textsc{Result}(\cdots \textsc{Result}(\textsc{Result}(I,\ a_1),\ a_2) \cdots,\ a_n) \in G$

Problem (Domain): $\langle S,\ A,\ \textsc{Actions},\ \textsc{Result},\ \textsc{Cost} \rangle$

$\textsc{Actions} : s \in S \rightarrow$

$\{\ a \in A : a \text{ can be executed (is applicable) in } s\ \}$

$\textsc{Result} : s \in S\ ,\ a \in A \rightarrow$

$s' \in S$ s.t. s' is the result of performing a in s

$\textsc{Cost}$: Assigns a cost to each path/step $c(s, a, s')$

Problem (Instance): $\langle I \in S,\ G \subseteq S \rangle$

Solution: $\langle a_1,\ a_2,\ \ldots,\ a_n \rangle \in A^n$ s.t.

$\textsc{Result}(\cdots \textsc{Result}(\textsc{Result}(I,\ a_1),\ a_2) \cdots,\ a_n) \in G$

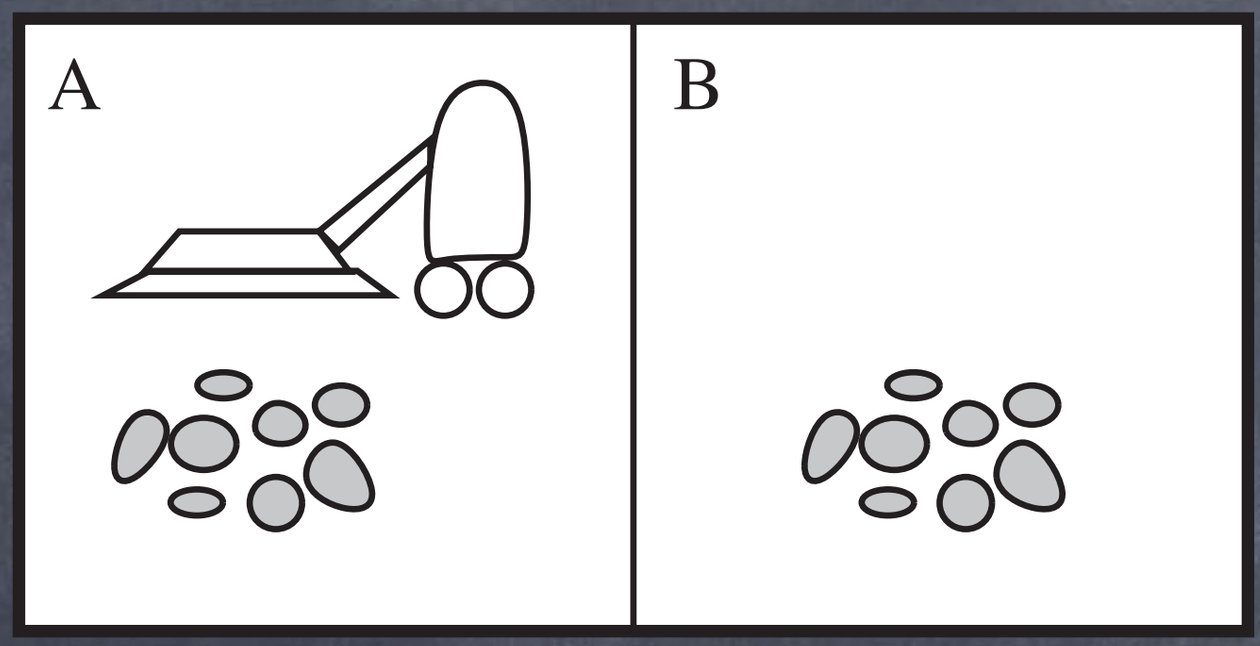
Romania Driving

- States: What city are we in
- Actions: Drive to another city
- Transition model: Adjacency per map
- Cost: sum of distances between cities
- Initial state: any city
- Goal states: any city

Romania Driving

- $S = \{$A,B,C,D,E,F,G,H,I,L,M,N,O,P,R,S,T,U,V,Z $\}$
- $A = \{$a,b,c,d,e,f,g,h,i,l,m,n,o,p,r,s,t,u,v,z $\}$
- Let ↑a=A, ↑b=B, ..., ↑z=Z
- ACTIONS(s) includes a iff s is adjacent to $\uparrow a$ on the map
- RESULT(s, a) = $\uparrow a$
- COST: $c(s, a, s') =$ map dist btw s and s'
- $I =$ any $x \in S$
- $G = \{ y \}$ for any $y \in S$

Vacuum World



Vacuum World

- States: Where is the vacuum, Room A clean/dirty, Room B clean/dirty
- Actions: Left, Right, Suck
- Transition model: move, clean, or no effect
- Cost: 1 per action
- Initial state: any
- Goal state: both rooms clean

Vacuum World

- $S = \{ \langle v, r_{\mathrm{A}}, r_{\mathrm{B}} \rangle \}$ where

  $v \in \{ \mathrm{A}, \mathrm{B} \}$ = location of vacuum

  $r_i \in \{ \mathit{dirty}, \mathit{clean} \}$ = room i is dirty/clean

Vacuum World

- $S = \{\ \langle v, r_1, r_2 \rangle\ \}$
- $A = \{\ \mathit{Left},\ \mathit{Right},\ \mathit{Suck}\ \}$

Vacuum World

- $S = \{\ \langle v, r_{\mathrm{A}}, r_{\mathrm{B}} \rangle\ \}$
- $A = \{\ \mathit{Left},\ \mathit{Right},\ \mathit{Suck}\ \}$
- $\textsc{Actions}(\langle v, r_{\mathrm{A}}, r_{\mathrm{B}} \rangle) = A$
- $\textsc{Result}(\langle v, r_{\mathrm{A}}, r_{\mathrm{B}} \rangle,\ a) = \langle \mathrm{A}, r_{\mathrm{A}}, r_{\mathrm{B}} \rangle$ if $a{=}\mathit{Left}$
  $\textsc{Result}(\langle v, r_{\mathrm{A}}, r_{\mathrm{B}} \rangle,\ a) = \langle \mathrm{B}, r_{\mathrm{B}}, r_{\mathrm{B}} \rangle$ if $a{=}\mathit{Right}$
  $\textsc{Result}(\langle v, r_{\mathrm{A}}, r_{\mathrm{B}} \rangle,\ \mathit{Suck}) = \langle v, \mathit{clean}, r_{\mathrm{B}} \rangle$ if $v{=}\mathrm{A}$
  $\langle v, r_{\mathrm{A}}, \mathit{clean} \rangle$ if $v{=}\mathrm{B}$

Vacuum World

- $S = \{ \langle v, r_{\mathrm{A}}, r_{\mathrm{B}} \rangle \}$
- $A = \{ \mathit{Left}, \mathit{Right}, \mathit{Suck} \}$
- $\textsc{Actions}(\langle v, r_{\mathrm{A}}, r_{\mathrm{B}} \rangle) = A$
- $\textsc{Result}(\langle v, r_{\mathrm{A}}, r_{\mathrm{B}} \rangle, a) = \langle \mathrm{A}, r_{\mathrm{A}}, r_{\mathrm{B}} \rangle$ if $a{=}\mathit{Left}$
  $\textsc{Result}(\langle v, r_{\mathrm{A}}, r_{\mathrm{B}} \rangle, a) = \langle \mathrm{B}, r_{\mathrm{A}}, r_{\mathrm{B}} \rangle$ if $a{=}\mathit{Right}$
  $\textsc{Result}(\langle v, r_{\mathrm{A}}, r_{\mathrm{B}} \rangle, \mathit{Suck}) = \langle v, \mathit{clean}, r_{\mathrm{B}} \rangle$ if $v{=}\mathrm{A}$
  $\langle v, r_{\mathrm{A}}, \mathit{clean} \rangle$ if $v{=}\mathrm{B}$
- $\textsc{Cost}$: $c(s, a, \mathrm{s}') = 1$
- $I = \langle v, r_{\mathrm{A}}, r_{\mathrm{B}} \rangle$ for any v, r_{A}, r_{B}
- $G = \{ \langle v, \mathit{clean}, \mathit{clean} \rangle \}$ for any v

15-Puzzle


1 2 3 4
5 6 7 8
9 10 11 12
13 15 14


ThinkFun
Fifteen Puzzle
Fifteen Puzzle

8-Puzzle

7	2	4
5		6
8	3	1

	1	2
3	4	5
6	7	8

8-Puzzle

- States: 3x3 grid of numbers 0-8 (0 for blank)
- Actions: Move a tile into the blank
  alternative: Move the blank
- Transition model: Swap moved tile with blank
- Cost: 1 per action
- Initial state: any
- Goal state: tiles in order 0-8 (or whatever)

8-Puzzle

- $S = \{\ M_k\ \}$ where $M_k = [m_{i,j}]$ for $1 \leq i,j \leq 3,\ m_{i,j} \in \{0,\ldots,8\},\ m_{i,j}$ unique

8-Puzzle

- $S = \{\ M_k\ \}$ where $M_k = [m_{i,j}]$ ...
- $A = \{\ 1, ..., 8\ \}$

8-Puzzle

- $S = \{\ M_k\ \}$ where $M_k = [m_{i,j}]$ ...
- $A = \{\ 1, ..., 8\ \}$
- Let $loc(M,i) = \langle x,y \rangle$ be such that $m_{x,y} = i$ (location of tile i in state M)
- $\textsc{Actions}(M)$ includes a iff

  $loc(M,a) = \langle x_a, y_a \rangle$ and $loc(M,0) = \langle x_0, y_0 \rangle$

  and $|x_a - x_0| = 1$ and $|y_a - y_0| = 0$ or

  $|x_a - x_0| = 0$ and $|y_a - y_0| = 1$

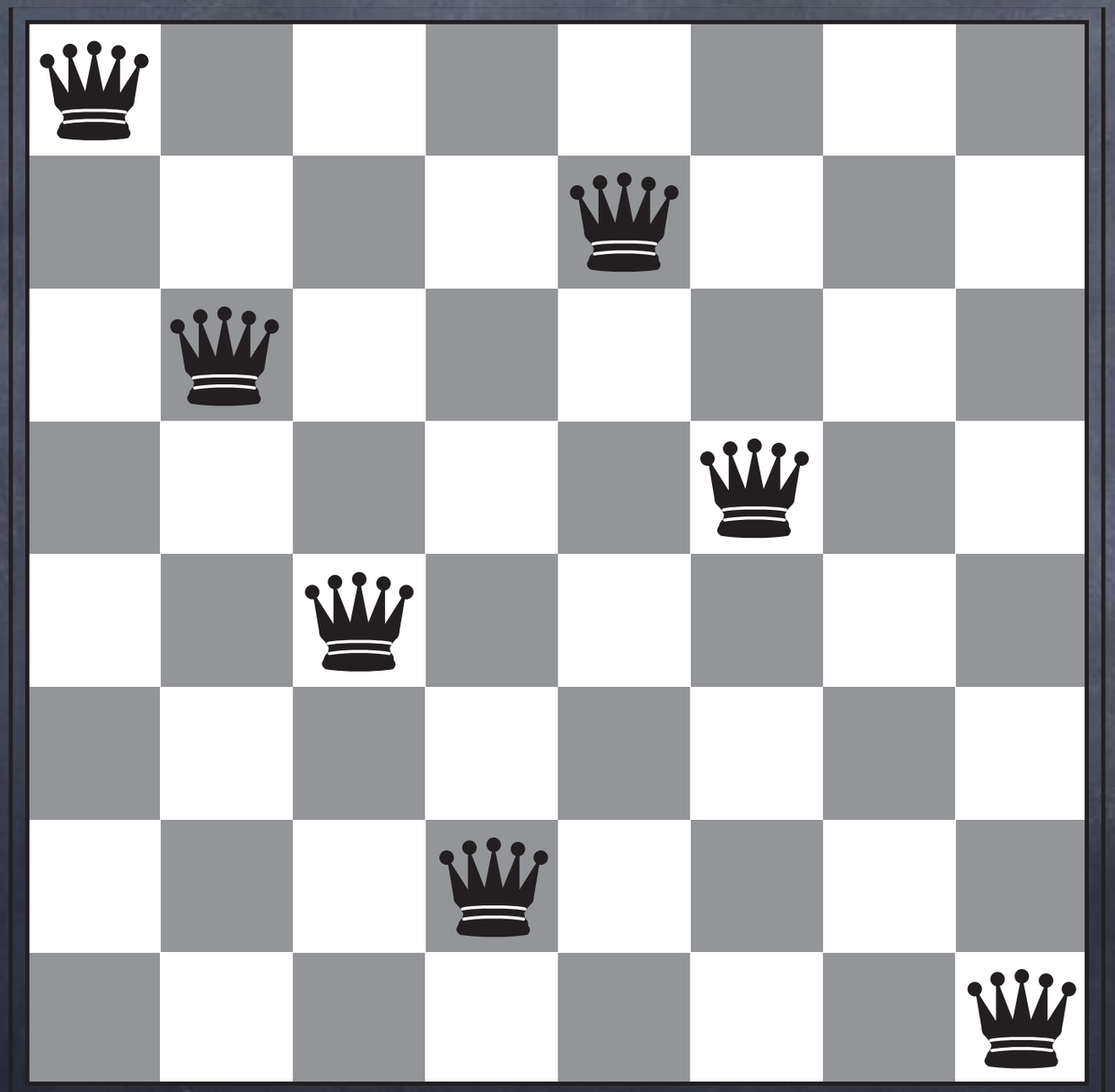
8-Puzzle

- $S = \{\ M_k\ \}$ where $M_k = [m_{i,j}]$ ...
- $A = \{\ 1, \ldots, 8\ \}$
- Let $loc(M,i) = \langle x,y \rangle$ ...
- $\textsc{Actions}(M)$ ...
- $\textsc{Result}(M,\ a)$:
  Let $loc(M,a) = \langle x_a, y_a \rangle$ and $loc(M,0) = \langle x_0, y_0 \rangle$
  Then in $\textsc{Result}(M,a)$: $m_{x0,y0} = a$ and $m_{xa,ya} = 0$

8-Puzzle

- $S = \{\ M_k\ \}$ where $M_k = [m_{i,j}]$ ...
- $A = \{\ 1, ..., 8\ \}$
- Let $loc(M,i) = \langle x,y \rangle$ ...
- $\textsc{Actions}(M)$ ...
- $\textsc{Result}(M,\ a)$ ...
- $\textsc{Cost}$: $c(s,a,s') = 1$
- $I =$ any M_k
- $G = \{\ [0, ..., 8]\ \}$ (or whatever)

8-Queens



Knuth's Conjecture

$$\left\lfloor \sqrt{\sqrt{\sqrt{\sqrt{\sqrt{(4!)!}}}}} \right\rfloor = 5$$

Problem (Domain): $\langle S,\ A,\ \textsc{Actions},\ \textsc{Result},\ \textsc{Cost} \rangle$

$\textsc{Actions} : s \in S \rightarrow$

$\{\ a \in A : a \text{ can be executed (is applicable) in } s\ \}$

$\textsc{Result} : s \in S\ ,\ a \in A \rightarrow$

$s' \in S$ s.t. s' is the result of performing a in s

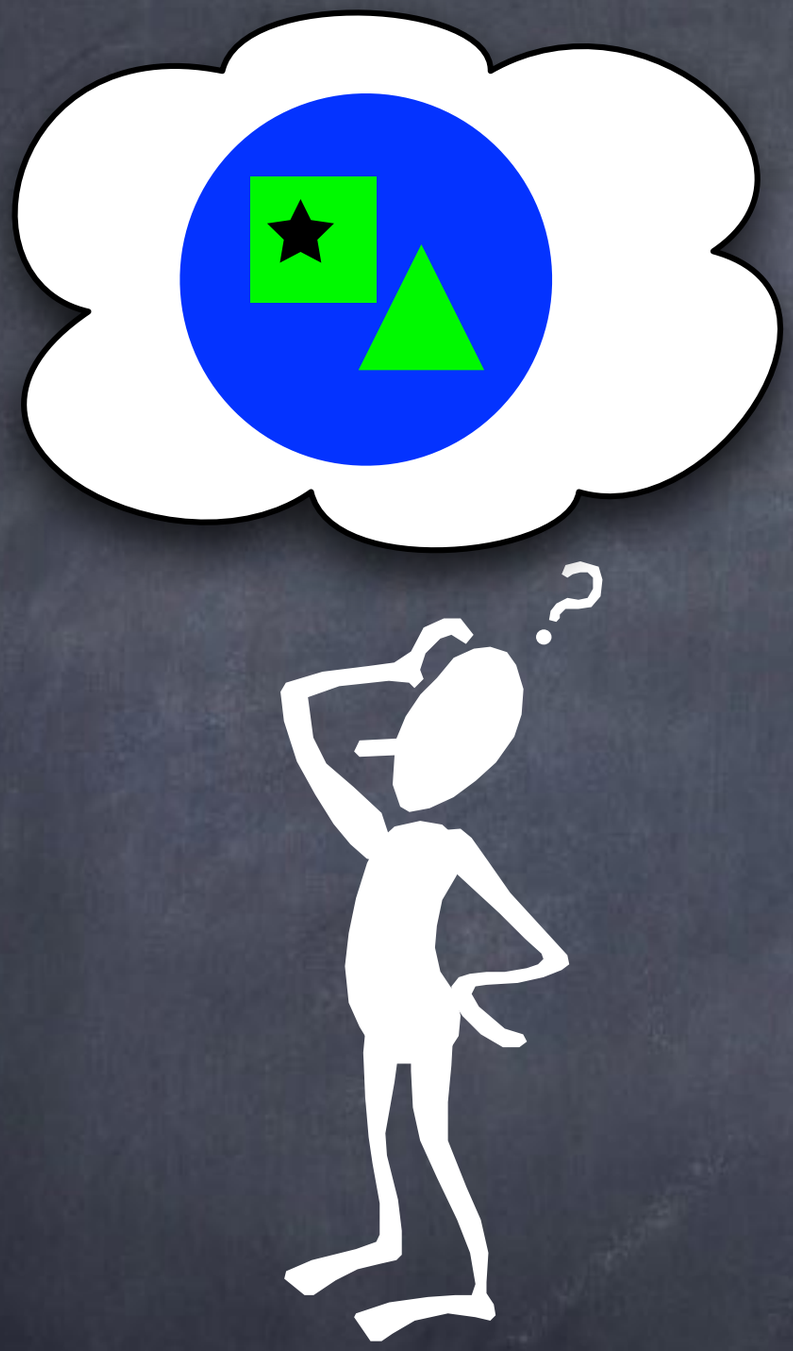
$\textsc{Cost}$: Assigns a cost to each path/step $c(s, a, s')$

Problem (Instance): $\langle I \in S,\ G \subseteq S \rangle$

Solution: $\langle a_1,\ a_2,\ \ldots,\ a_n \rangle \in A^n$ s.t.

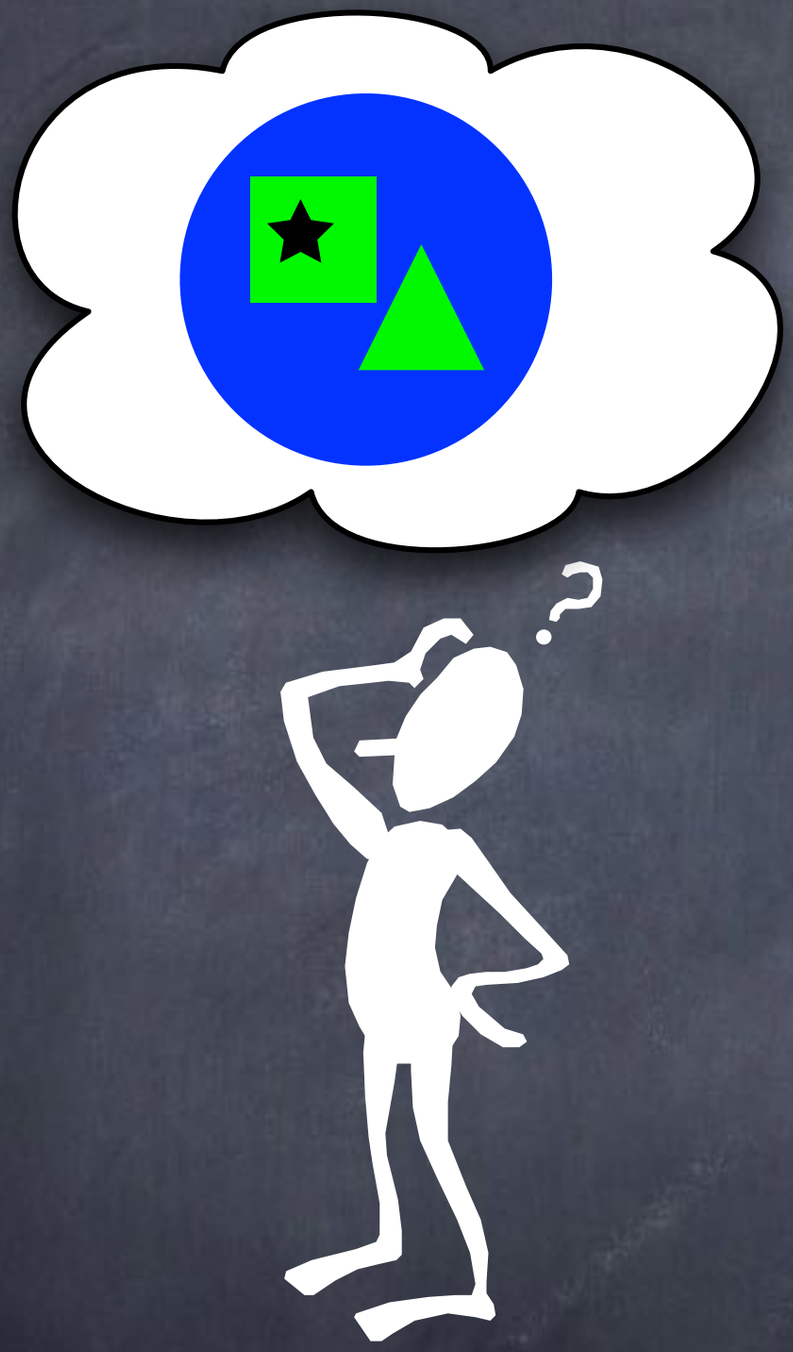
$\textsc{Result}(\cdots \textsc{Result}(\textsc{Result}(I,\ a_1),\ a_2) \cdots,\ a_n) \in G$

World state



Problem-solving state

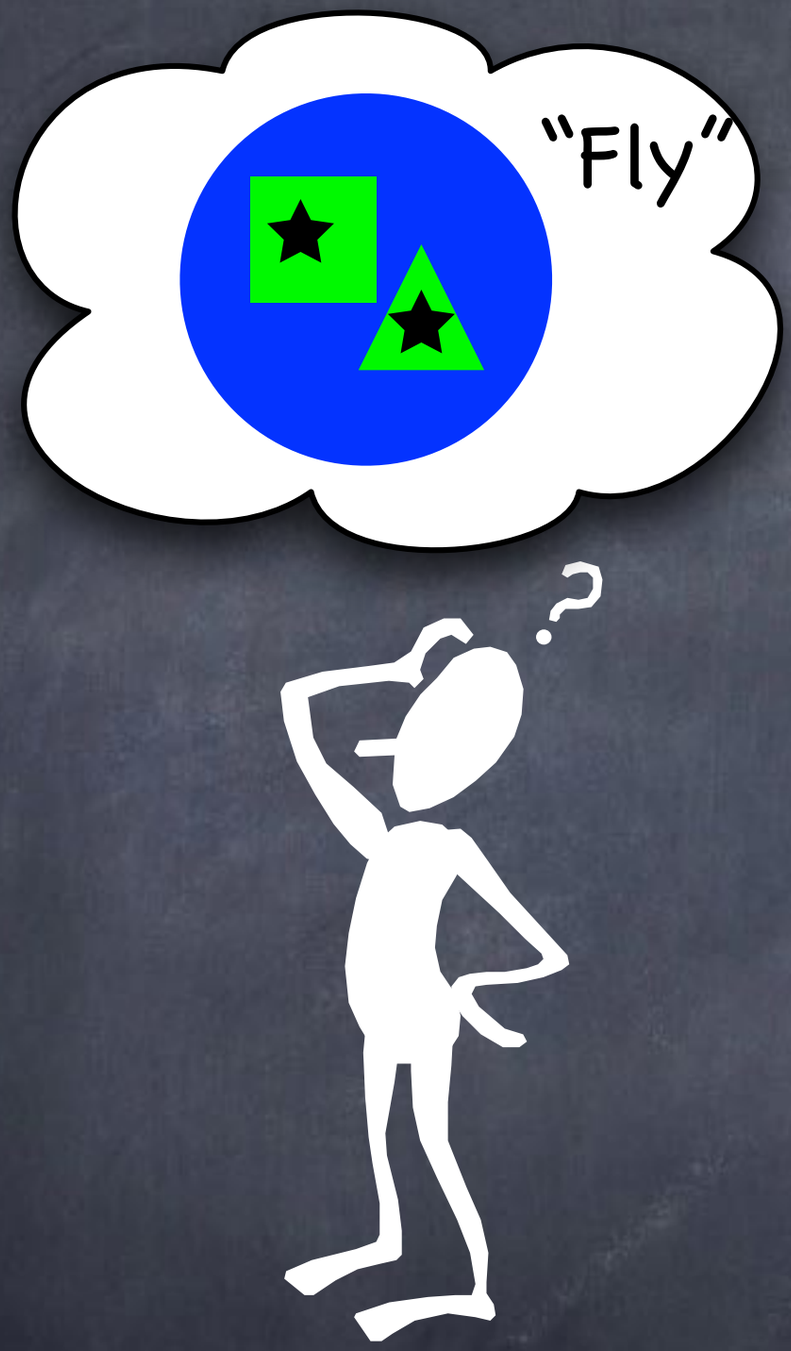
World state



Problem-solving state

World state

Action



Problem-solving state

Transition Model

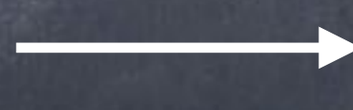
World state

Action

State Space

- The set of all states reachable from the initial state by some sequence of actions

States
Actions
Transition Model



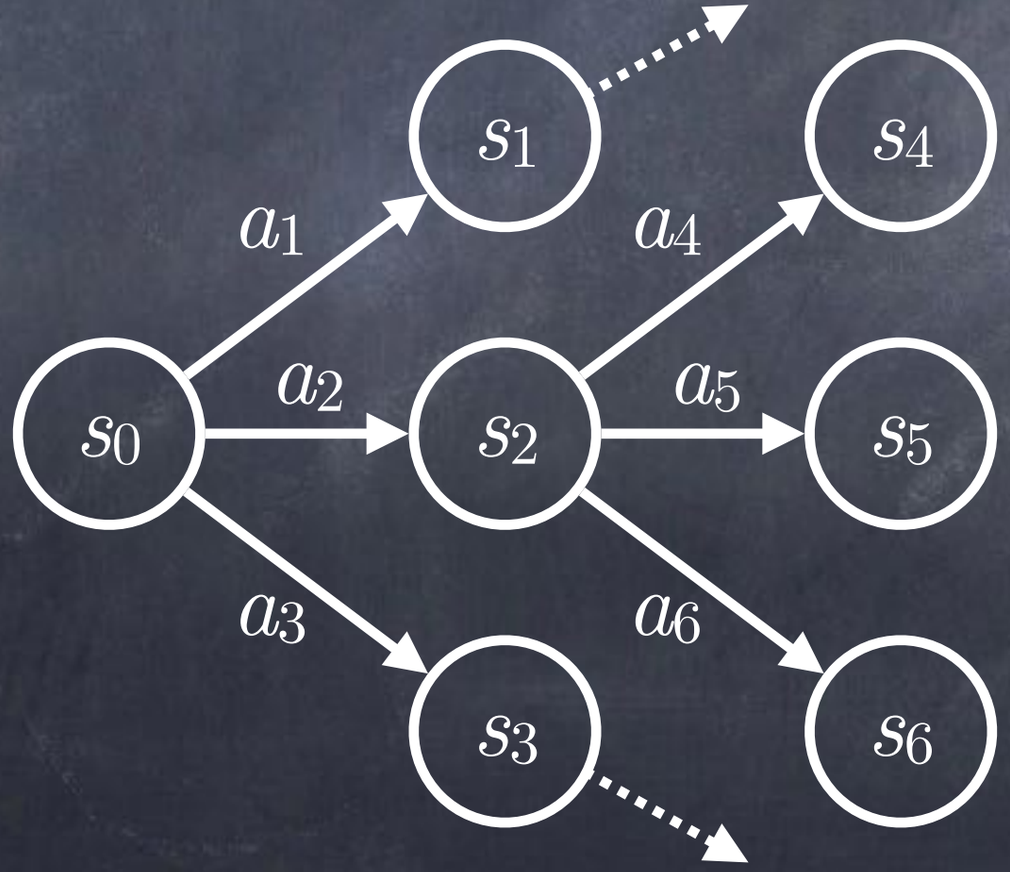
State Space

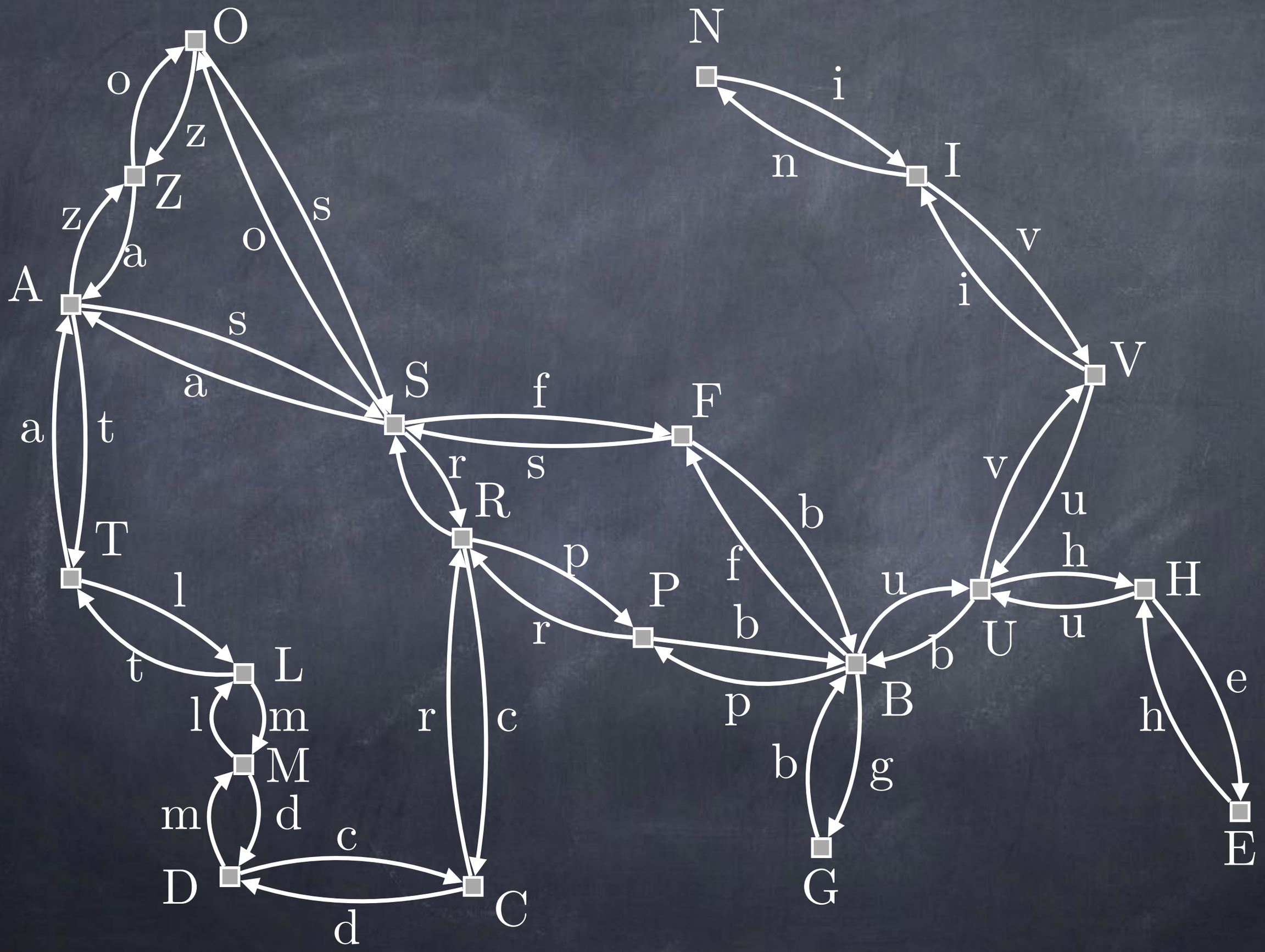
State Space

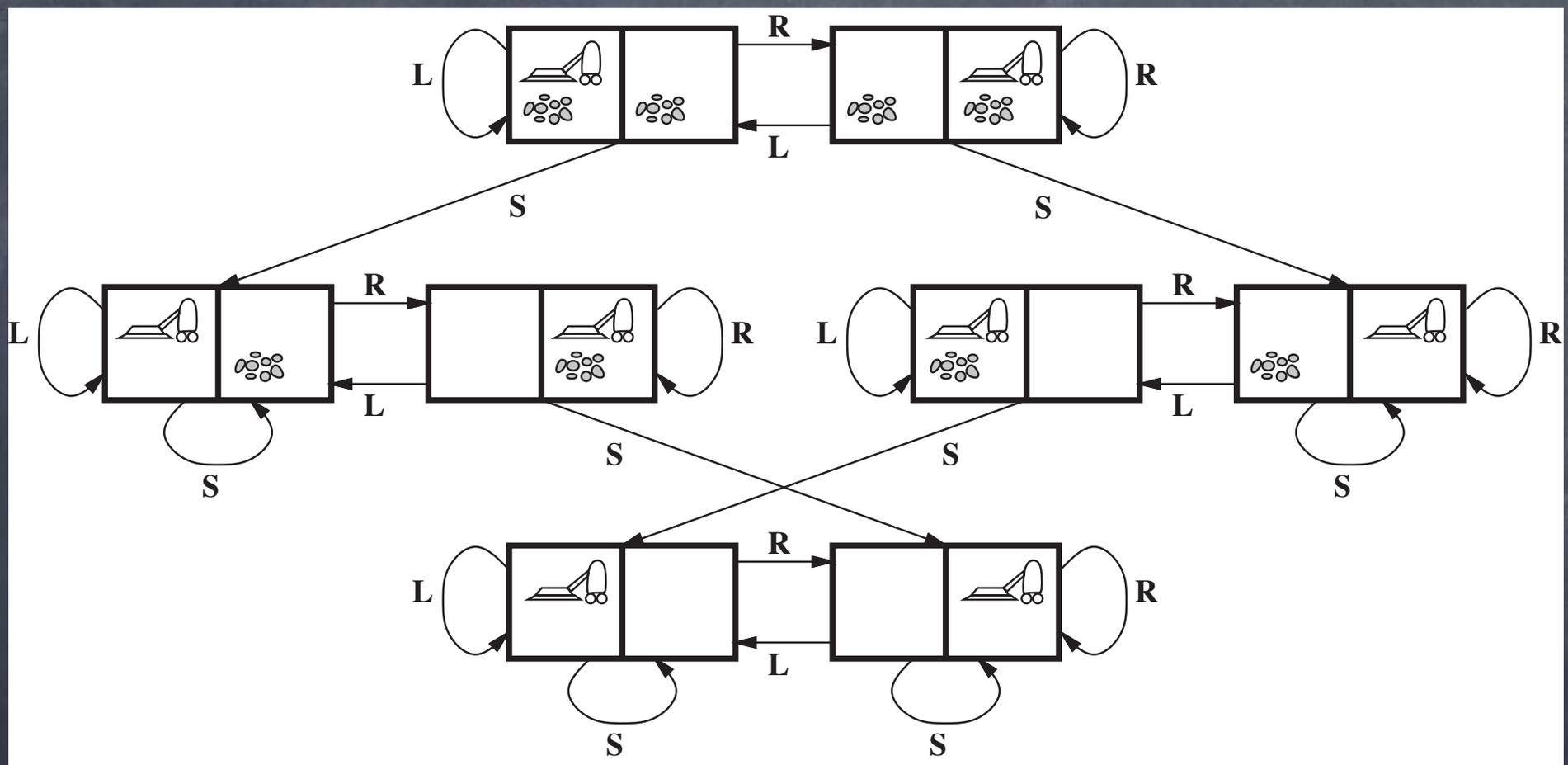
$\langle V, E \rangle$:

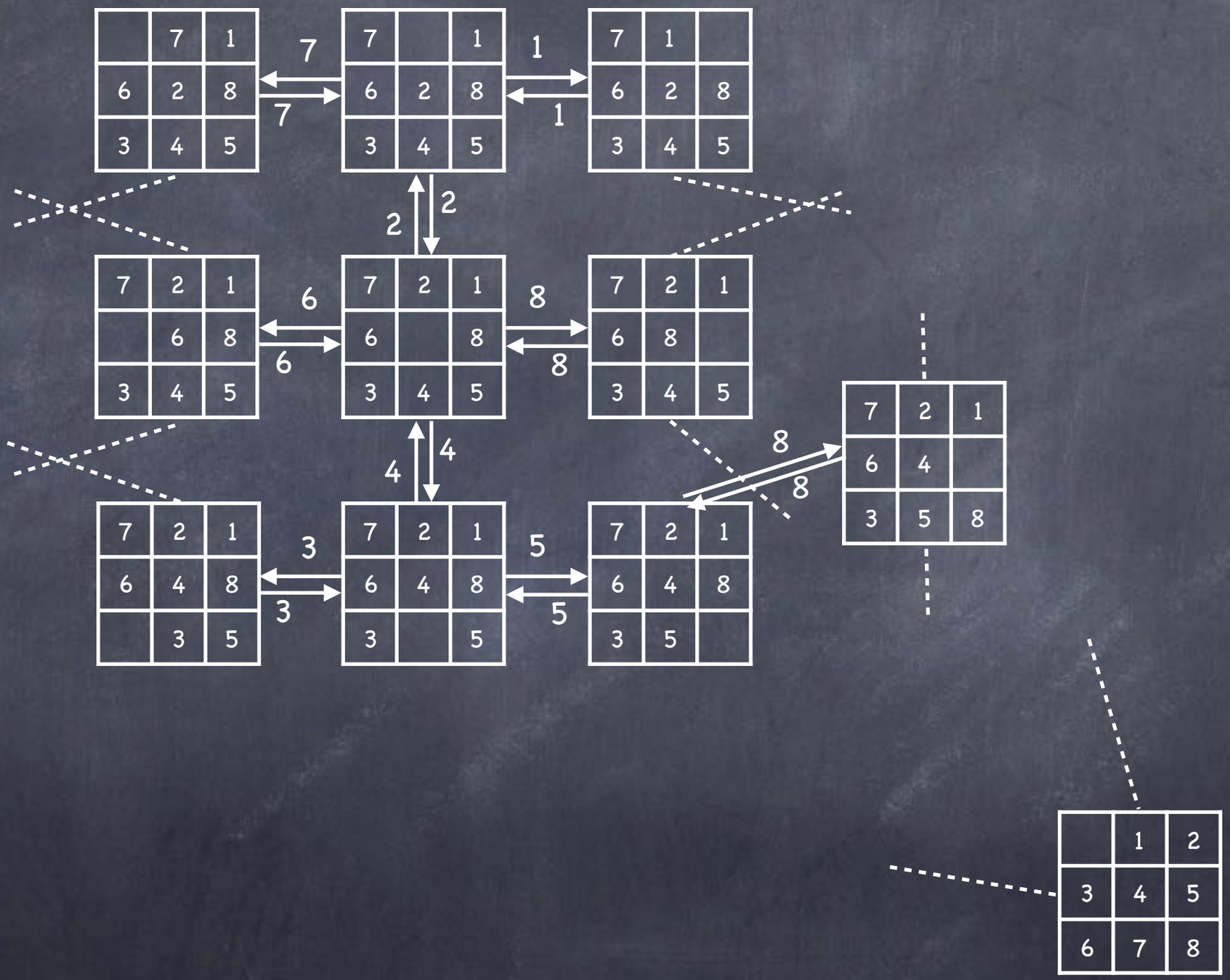
$V = \{ v_i \mid s_i \in \mathrm{S} \}$

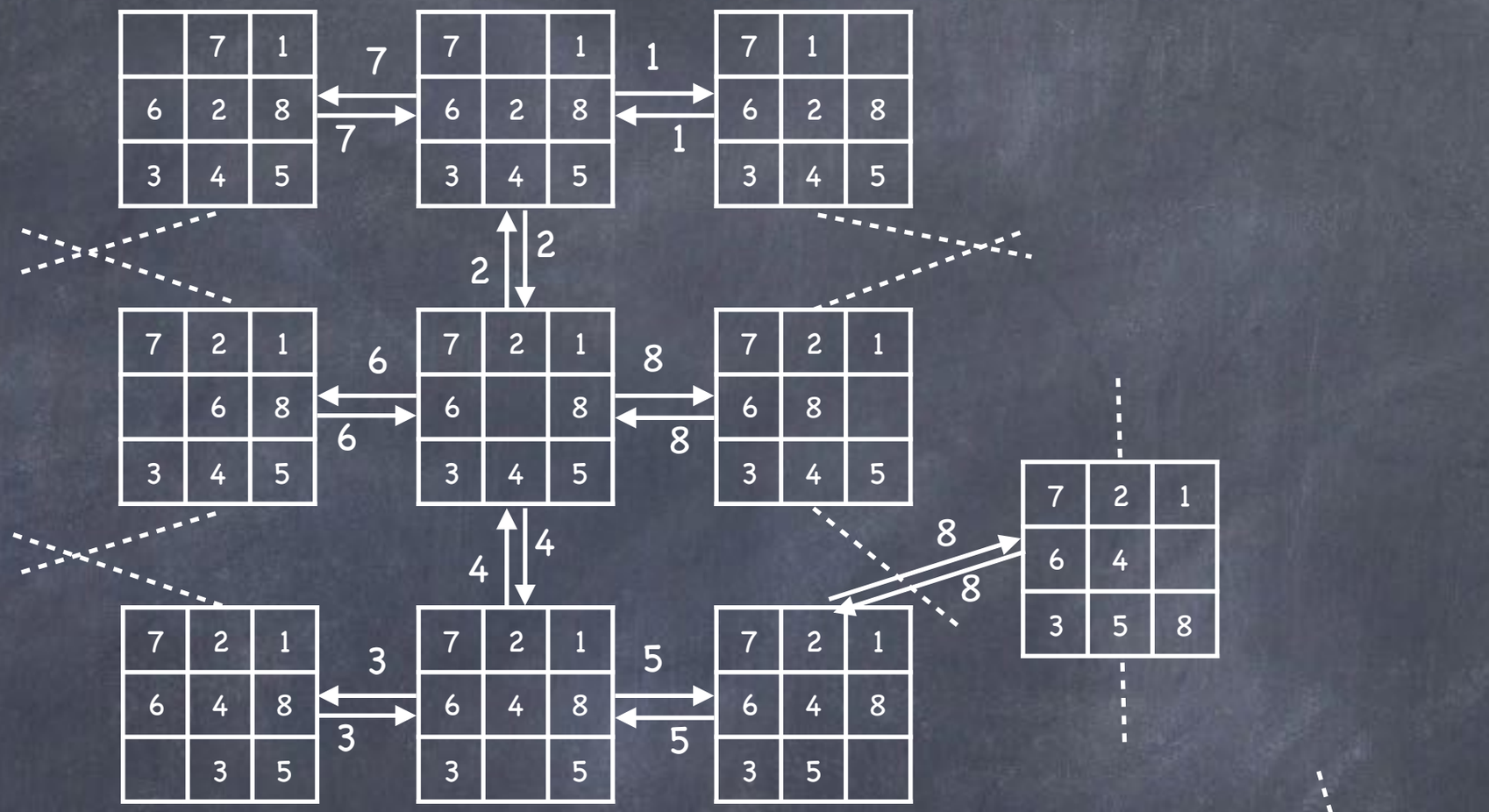
$\mathrm{E} = \{ \langle v_i, v_j, a \rangle \mid s_j = \textsc{Result}(s_i, a) \}$




O
o
z
Z
z
a
A
o
s
s
a
S
f
F
a
t
r
s
R
T
l
p
P
f
b
b
u
u
h
H
t
L
r
c
r
b
u
U
b
l
m
M
p
B
h
e
m
d
b
g
c
D
C
G
E
d
N
i
n
I
v
i
V
v
u


R
L
L
R
S
S
R
L
R
L
R
L
R
L
S
S
S
S
R
L
L
R
S
S

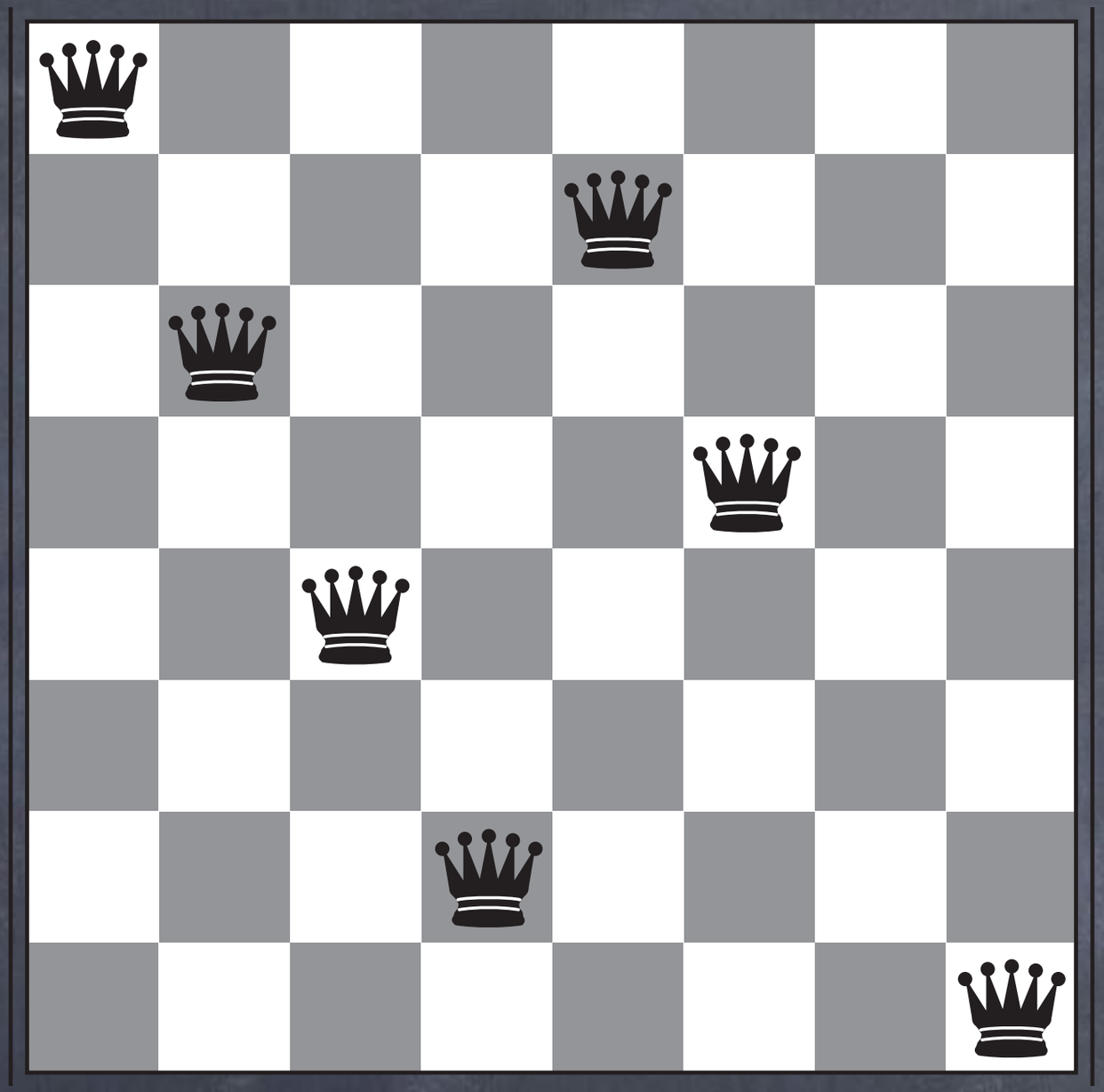

7 1
6 2 8
3 4 5
7
7
7 1
6 2 8
3 4 5
1
1
7 1
6 2 8
3 4 5
2
2
7 2 1
6 8
3 4 5
6
6
7 2 1
6 8
3 4 5
8
8
7 2 1
6 8
3 4 5
4
4
7 2 1
6 4 8
3 5
3
3
7 2 1
6 4 8
3 5
5
5
7 2 1
6 4 8
3 5
8
8
7 2 1
6 4
3 5 8
1 2
3 4 5
6 7 8



8-puzzle: $9!/2 = 181{,}440$

15-puzzle: $16!/2 = \sim 1.3$ trillion

24-puzzle: $25!/2 = \sim 10^{25}$



8-queens: $63 \times 62 \times ... \times 57 = \tilde{}1.8 \times 10^{14}$

Can be improved...

$$\left\lfloor \sqrt{\sqrt{\sqrt{\sqrt{\sqrt{(4!)!}}}}} \right\rfloor = 5$$

State space is infinite
(as far as we know)

State-Based Model

- Model the world as a set of states
- Actions change the state of the world

Abstraction!

- State Space
  - Defined by states, actions, and transition model
  - Small, large, or even infinite

Romania

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

Romania

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

Romania

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

Romania

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

Romania

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

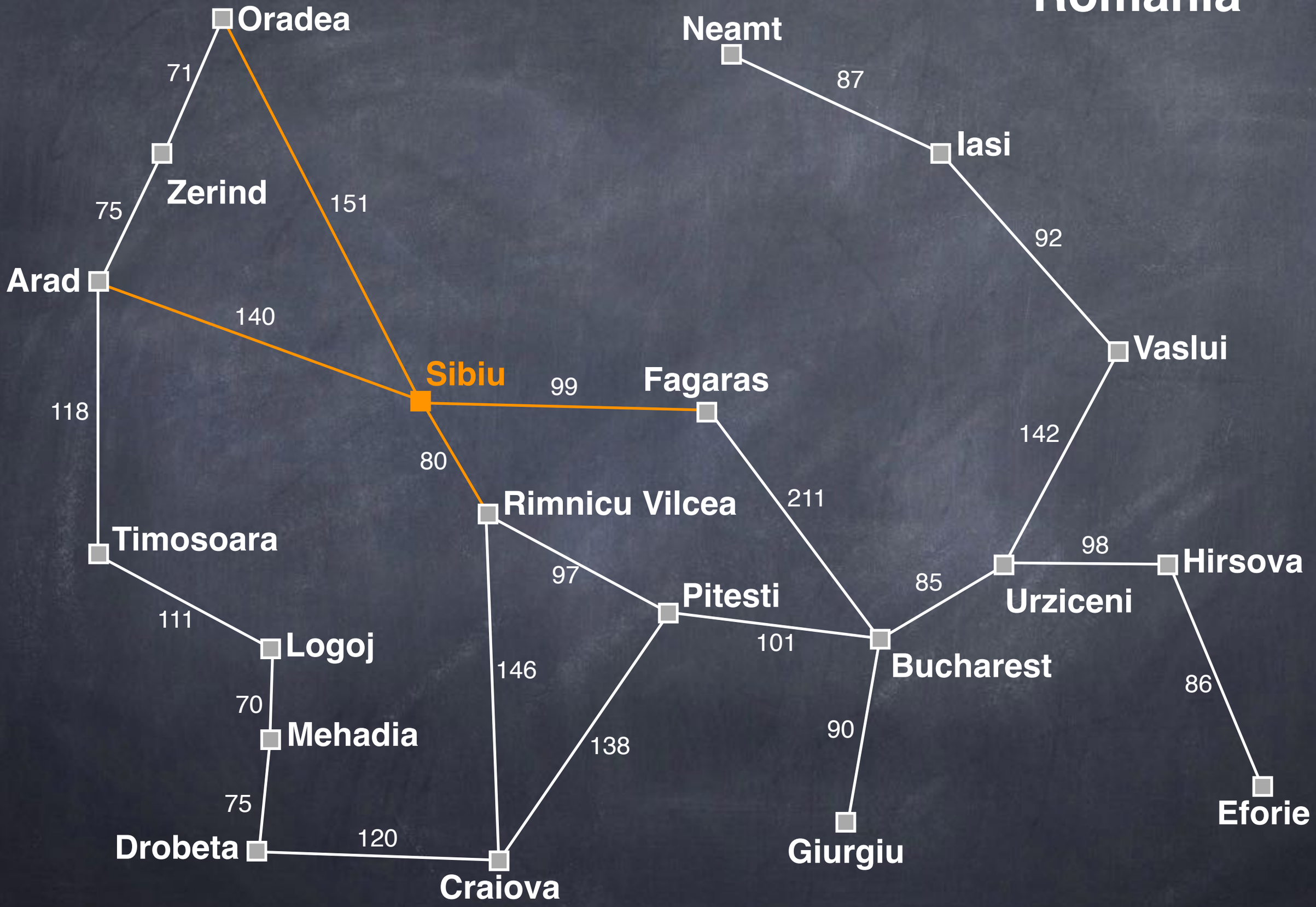
Romania

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

Romania

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

Romania


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

Romania

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

Romania

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

Romania

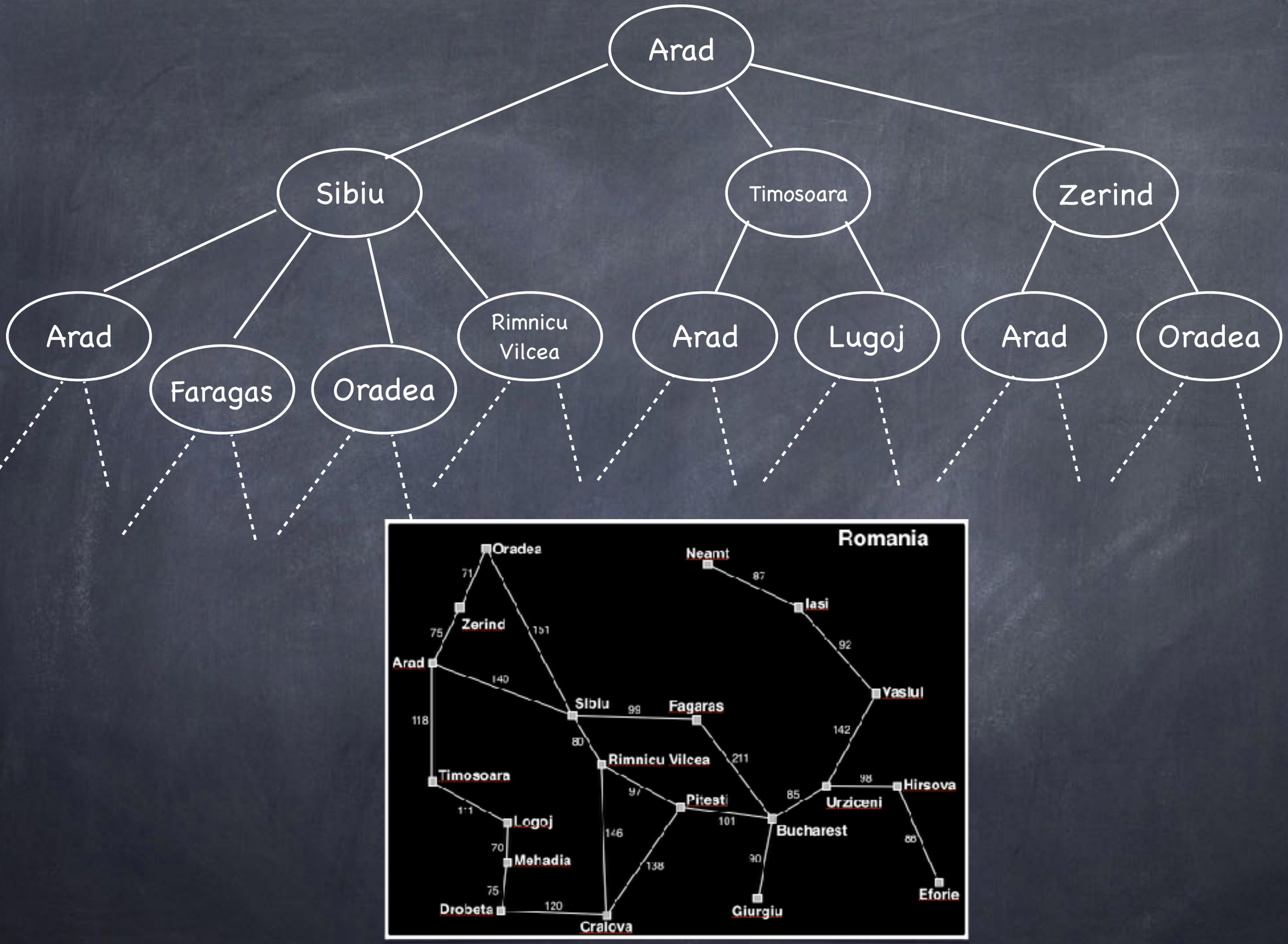
Oradea
71
Zerind
75
151
Arad
140
Sibiu
99
Fagaras
118
80
Rimnicu Vilcea
211
Timosoara
97
Pitesti
111
Logoj
101
146
Bucharest
70
Mehadia
90
138
75
Drobeta
120
Craiova
Giurgiu
Neamt
87
Iasi
92
Vaslui
142
98
Hirsova
85
Urziceni
86
Eforie

Romania

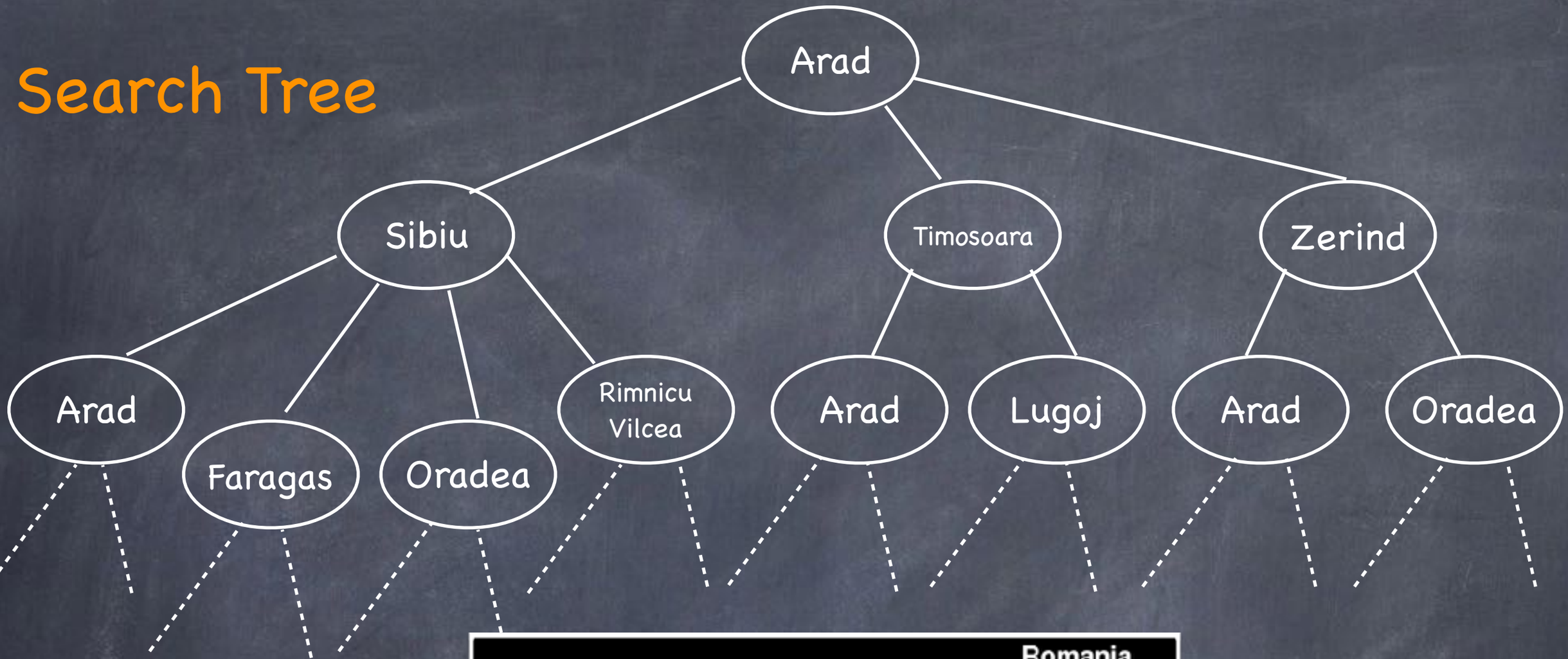
Oradea
71
Neamt
87
Zerind
75
151
Iasi
92
Arad
140
Vaslui
Sibiu
99
Fagaras
118
142
80
Rimnicu Vilcea
211
Timosoara
98
Hirsova
97
Pitesti
85
Urziceni
111
101
Logoj
Bucharest
146
86
70
Mehadia
90
138
75
Eforie
Drobeta
120
Giurgiu
Craiova

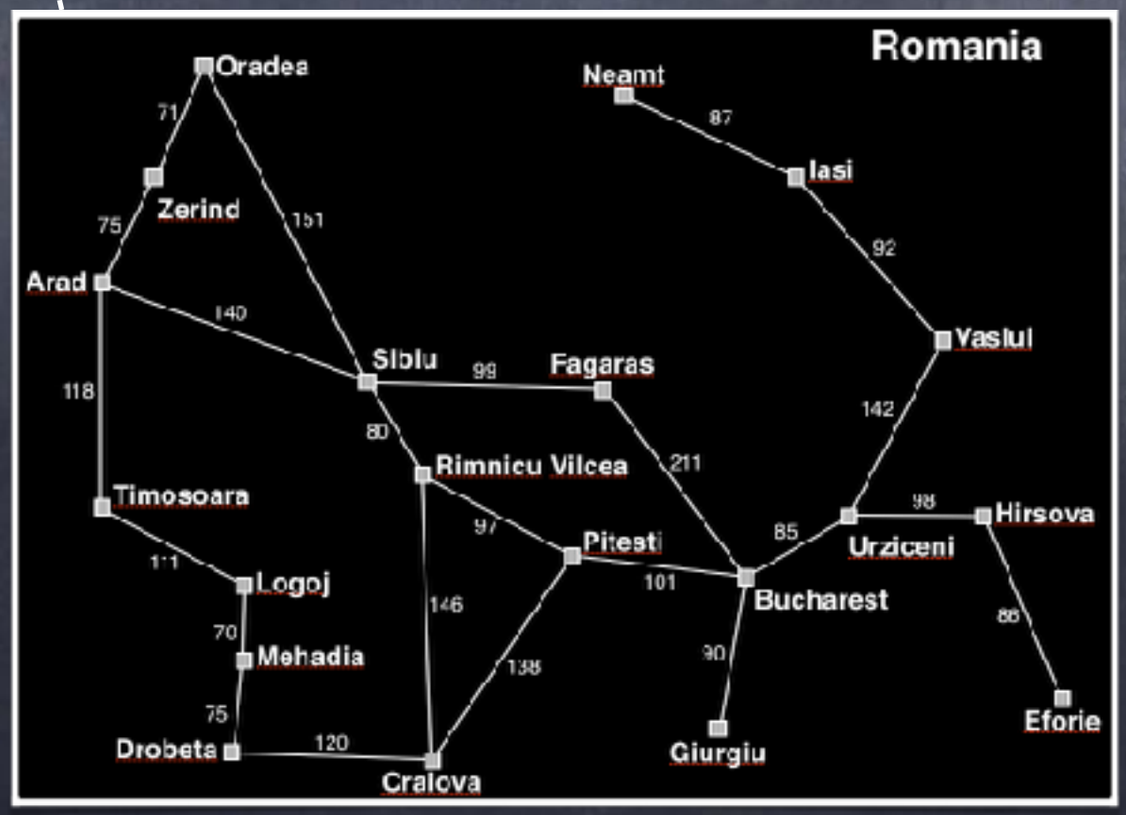
Romania

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

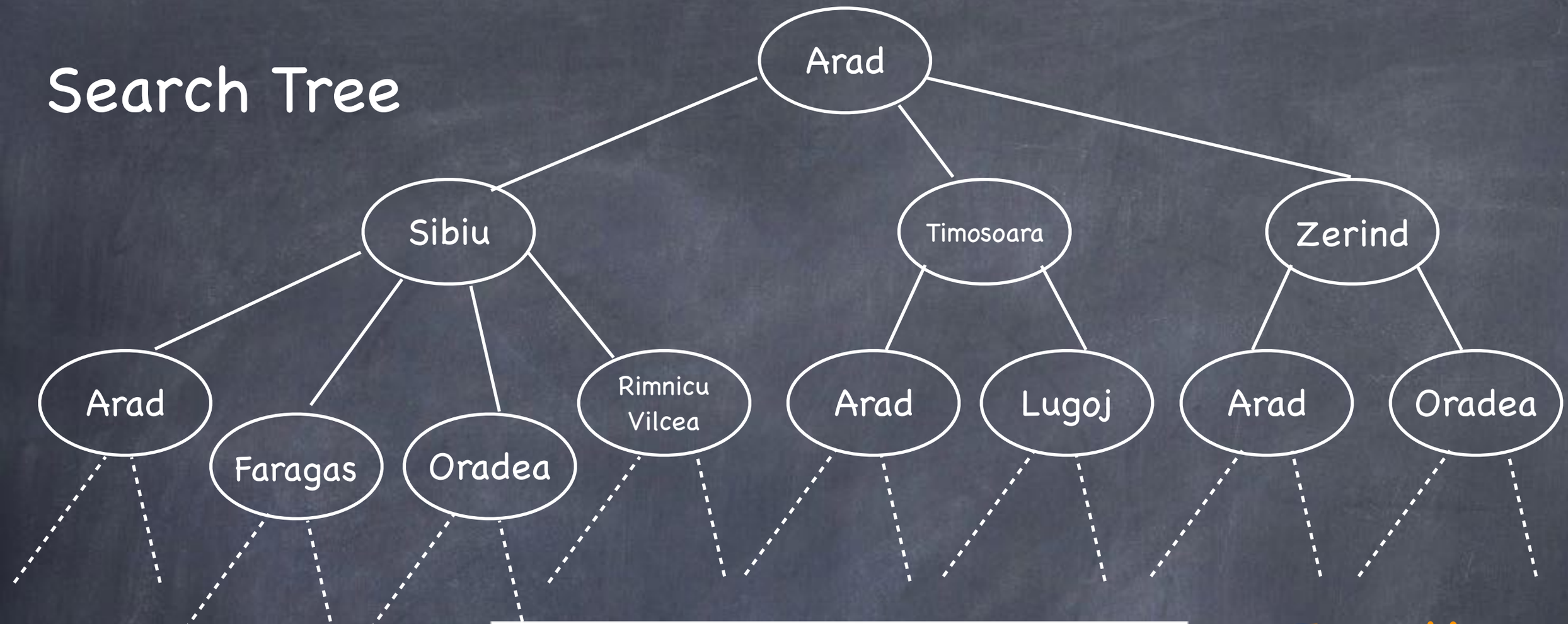

Arad
Sibiu
Timosoara
Zerind
Arad
Faragas
Oradea
Rimnicu Vilcea
Arad
Lugoj
Arad
Oradea
Romania
Oradea
Neamt
Zerind
Iasi
Arad
Yasiui
Siblu
Fagaras
Rimnicu Vilcea
Timosoara
Hirsova
Pitesti
Urziceni
Logoj
Bucharest
Mehadia
Eforie
Drobeta
Craiova
Giurgiu
71
87
75
151
92
140
118
99
142
80
211
98
97
85
111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

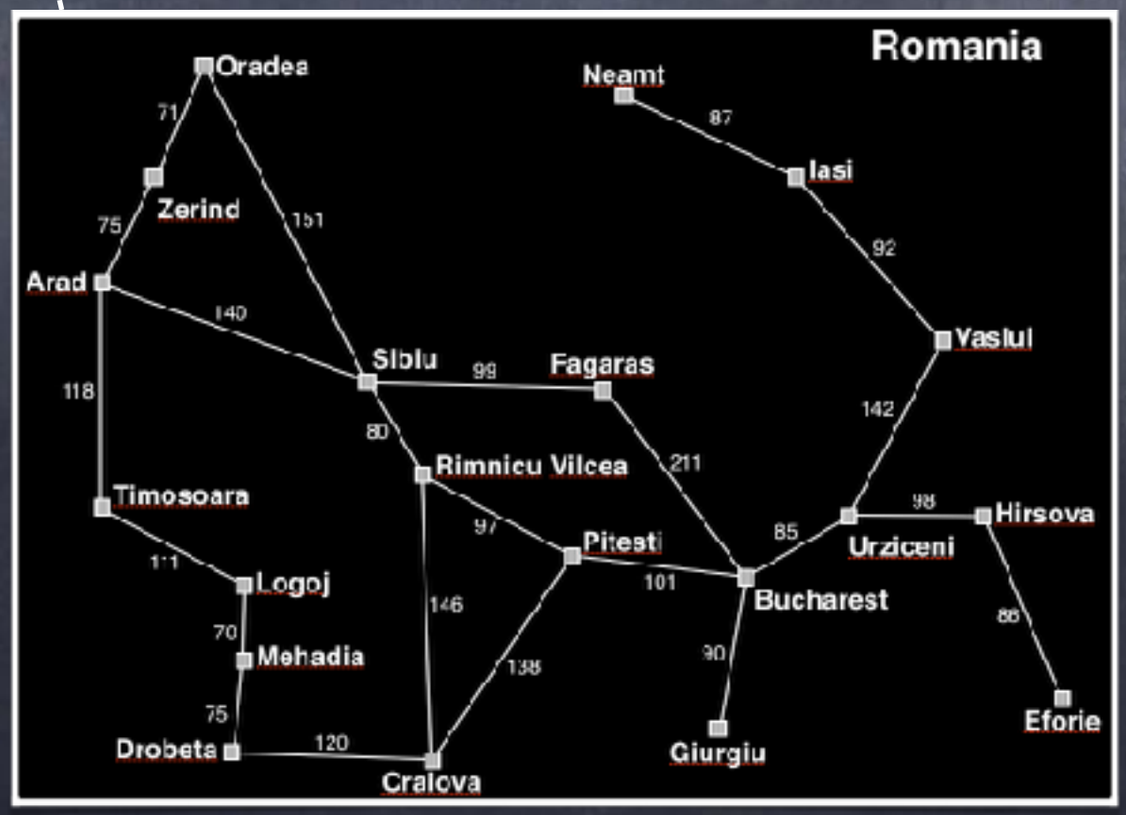
Search Tree


Arad
Sibiu
Timosoara
Zerind
Arad
Faragas
Oradea
Rimnicu Vilcea
Arad
Lugoj
Arad
Oradea


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

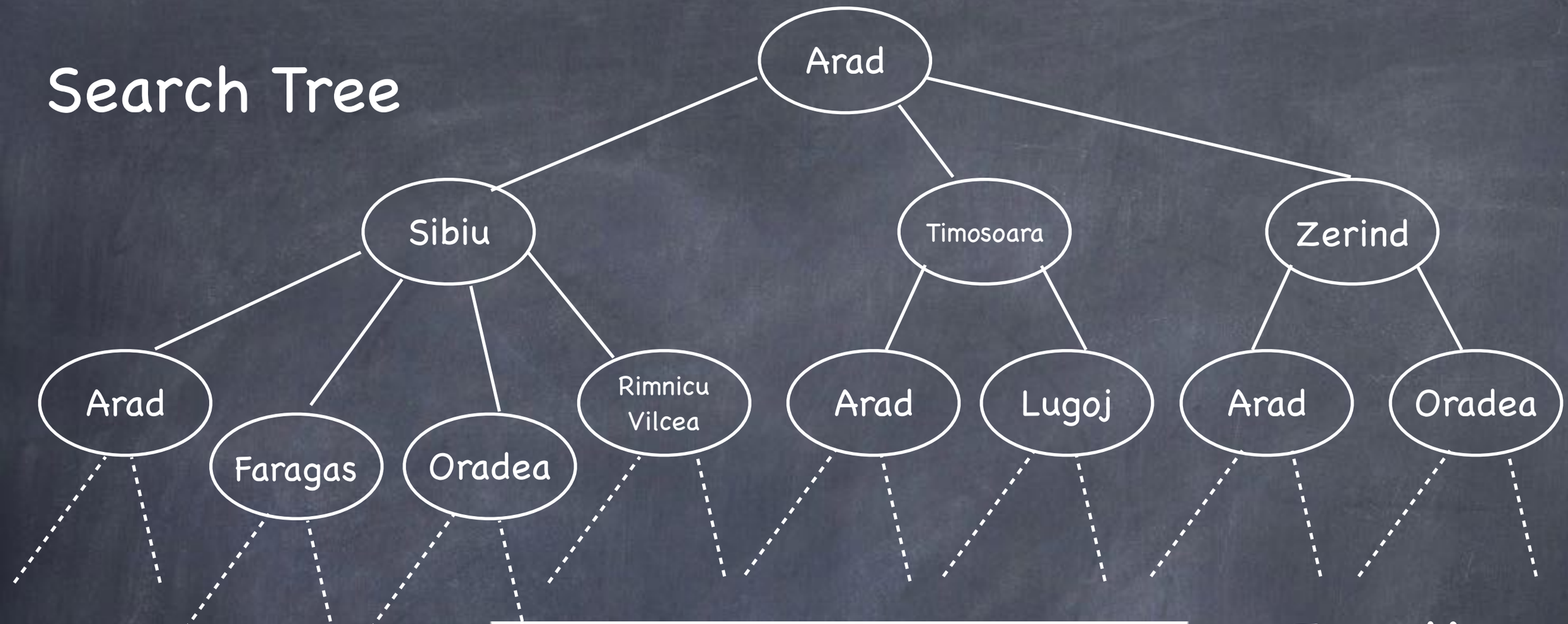
Search Tree

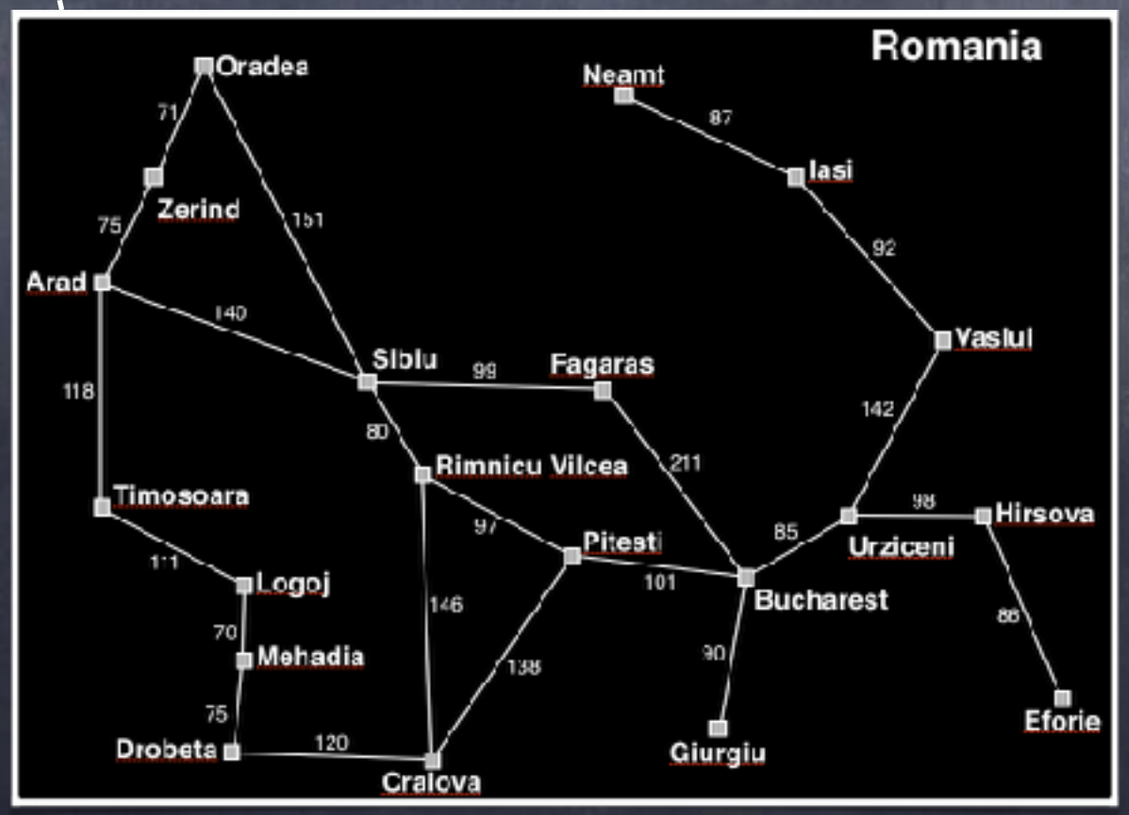




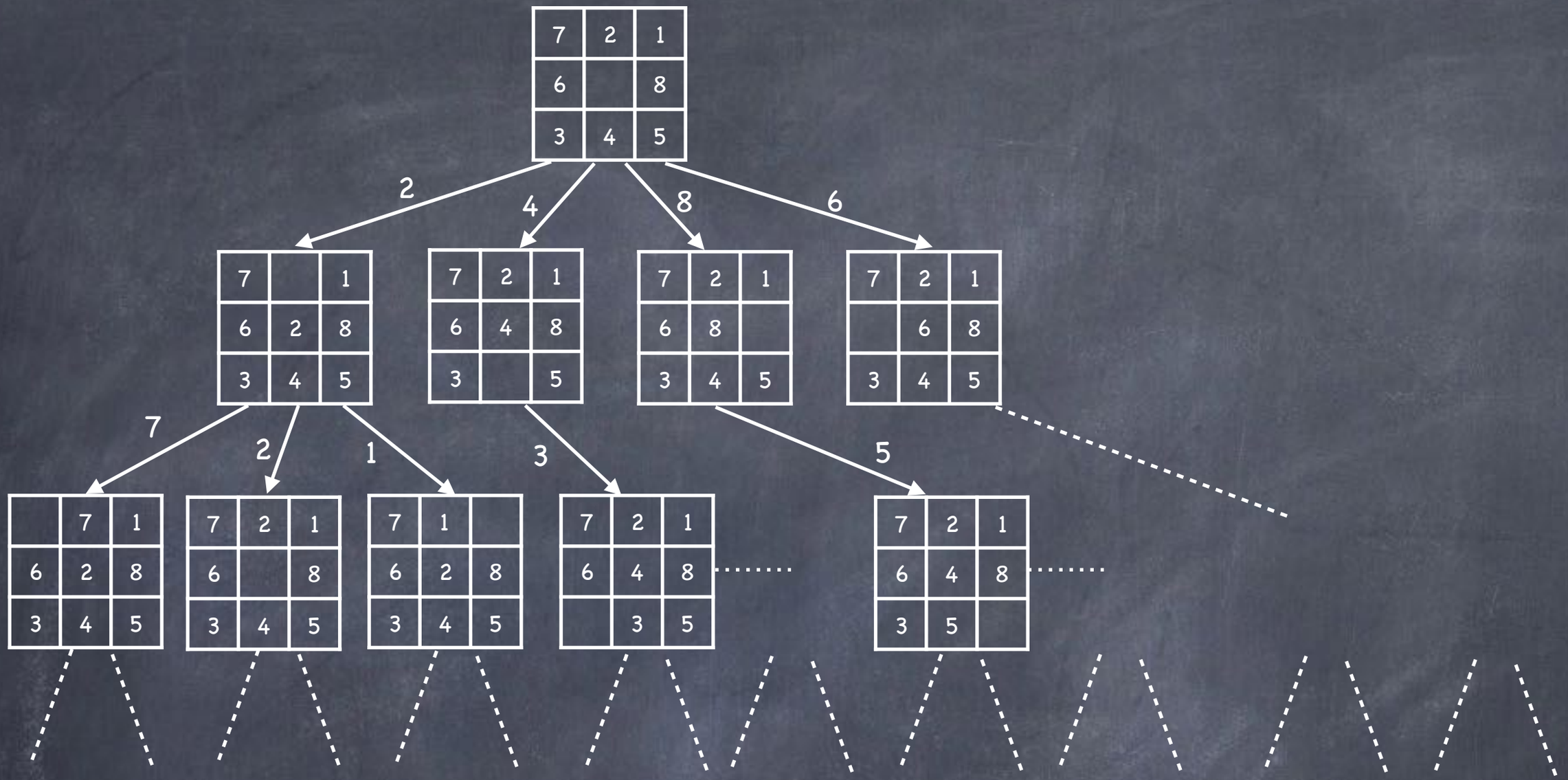
Frontier

Search Tree





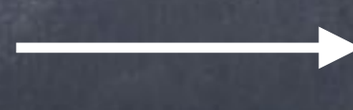
Frontier


7 2 1
6 8
3 4 5
2
4
8
6
7 1
6 2 8
3 4 5
7 2 1
6 4 8
3 5
7 2 1
6 8
3 4 5
7 2 1
6 8
3 4 5
7
2
1
3
5
7 1
6 2 8
3 4 5
7 2 1
6 8
3 4 5
7 1
6 2 8
3 4 5
7 2 1
6 4 8
3 5
7 2 1
6 4 8
3 5

State Space

- The set of all states reachable from the initial state by some sequence of actions

States
Actions
Transition Model



State Space

State-Space Search Tree

- The tree formed by starting with the initial state and using applicable actions to generate successor (child) states
- May be infinite (even for finite search spaces)
- Not stored explicitly

State-Space Search

- Start with initial state
- Generate successor states by applying applicable actions
- Until you find a goal state

Initialize the frontier to just I

While the frontier is not empty:

Remove a state s from the frontier

If $s \in G$:

Return solution to s

else:

Add $successors(s)$ to the frontier

Universal Problem-Solving Procedure

Initialize the frontier to just I

While the frontier is not empty:

Remove a state s from the frontier

If $s \in G$:

Return solution to s

else:

Add $successors(s)$ to the frontier

Coolest Program Ever

Initialize the frontier to just I
While the frontier is not empty:
Remove a state s from the frontier
If $s \in G$:
    Return solution to s
else:
    Add $successors(s)$ to the frontier

NO CODE REQUIRED!

AIMA Fig. 3.7

```
Solution treeSearch(Problem p) {
  Set<Node> frontier = new Set<Node>(p.getInitialState());

  while (true) {
    if (frontier.isEmpty()) {
      return null;
    }
    Node node = frontier.selectOne();
    if (p.isGoalState(node.getState())) {
      return n.getSolution();
    }

    for (Node n : node.expand()) {

      frontier.add(n);

    }
  }
}
```

AIMA Fig. 3.7

```
Solution graphSearch(Problem p) {
  Set<Node> frontier = new Set<Node>(p.getInitialState());
  Set<Node> explored = new Set<Node>();
  while (true) {
    if (frontier.isEmpty()) {
      return null;
    }
    Node node = frontier.selectOne();
    if (p.isGoalState(node.getState())) {
      return n.getSolution();
    }
    explored.add(node);
    for (Node n : node.expand()) {
      if (!explored.contains(n)) {
        frontier.add(n);
      }
    }
  }
}
```

THE OFFICIAL

METALLICA

ILLUSTRATED CHRONICLES

SO WHAT!

THE GOOD, THE MAD AND THE UGLY

EDITED BY STEFFAN CHIRAZI

Summary

- Formal model of problems and solutions based on states and actions that transition between them
- General-purpose algorithm for solving any problem that can be represented using the model
- State-space search framework will allow us to explore and compare different problem-solving strategies

State-Space Search

```
Solution graphSearch(Problem p) {
  Set<Node> frontier = new Set<Node>(p.getInitialState());
  Set<Node> explored = new Set<Node>();
  while (true) {
    if (frontier.isEmpty()) {
      return null;
    }
    Node node = frontier.selectOne();
    if (p.isGoalState(node.getState())) {
      return n.getSolution();
    }
    explored.add(node);
    for (Node n : node.expand()) {
      if (!explored.contains(n)) {
        frontier.add(n);
      }
    }
  }
}
```

For Next Time:

Finish Chapter 3