Probabilistic Reasoning

ARTIFICIAL INTELLIGENCE

JUCHEOL MOON

Bayesian networks

- A simple, graphical notation for conditional independence assertions and hence for compact specification of full joint distributions
- In the simplest case, conditional distribution represented as a conditional probability table (CPT) giving the distribution over X_i for each combination of parent values

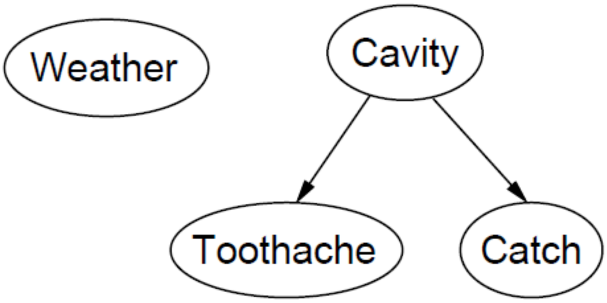
Bayesian networks

- Syntax:
  - a set of nodes, one per variable
  - a directed, acyclic graph
  - a conditional distribution for each node given its parents:

3

Example

- Topology of network encodes conditional independence assertions:
  - Weather is independent of the other variables
  - Toothache and Catch are conditionally independent given Cavity



4

Example

- I'm at work, neighbor John calls to say my alarm is ringing, but neighbor Mary doesn't call. Sometimes it's set off by minor earthquakes. Is there a burglar?
- Variables
- Network topology reflects "causal" knowledge

5

5

Example

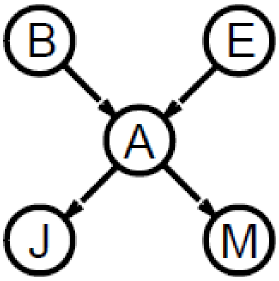
- Variables
  - Burglar, Earthquake, Alarm, JohnCalls, MaryCalls
- Network topology reflects "causal" knowledge:
  - A burglar can set the alarm on
  - An earthquake can set the alarm on
  - The alarm can cause Mary to call
  - The alarm can cause John to call

6

6

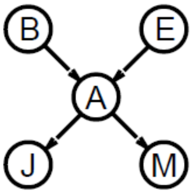
Compactness

- A CPT for Boolean X_i with k Boolean parents has ____ rows for the combinations of parent values
- Each row requires probability p for $X_i = true$
  - The probability for $X_i = false$ is __________



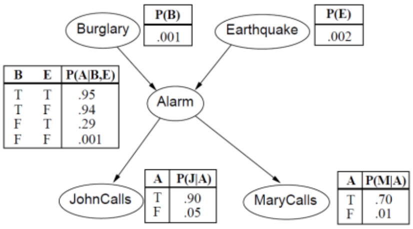
Representing the full joint distribution

- Chain rule
  - $\vec{P}(X_1, \cdots, X_n) =$
- Conditional independences
  - $\vec{P}(X_i | X_1, \cdots, X_{i-1}) =$
- Consequence
  - $\vec{P}(X_1, \cdots, X_n) =$



Representing the full joint distribution

- Global semantics defines the full joint distribution as the product of the local conditional distributions:



P(B)
.001

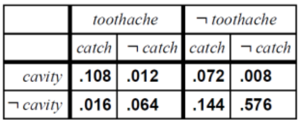
P(E)
.002

B	E	P(A|B,E)
T	T	.95
T	F	.94
F	T	.29
F	F	.001

A	P(J|A)
T	.90
F	.05

A	P(M|A)
T	.70
F	.01

General inference procedure.

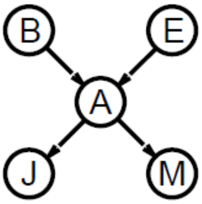


	toothache		*¬ toothache*	
	catch	*¬ catch*	*catch*	*¬ catch*
cavity	**.108**	**.012**	**.072**	**.008**
¬ cavity	**.016**	**.064**	**.144**	**.576**

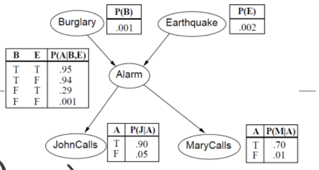
- $\vec{P}(X|e) = \alpha\vec{P}(X,e) = \alpha\sum_{y\in Y}\vec{P}(X,e,y)$
  - The query involves a single variable, X
    - Cavity in the example
  - $\vec{E}$ be the list of evidence variables
    - Toothache in the example
    - $\vec{e}$ be the list of observed values for them
  - $\vec{Y}$ be the remaining unobserved variables
    - Catch in the example

Inference by enumeration

- Simple query on the burglary network:
  - $\vec{P}(B|j,m)$



Inference by enumeration



P(B)
.001

P(E)
.002

B	E	P(A|B,E)
T	T	.95
T	F	.94
F	T	.29
F	F	.001

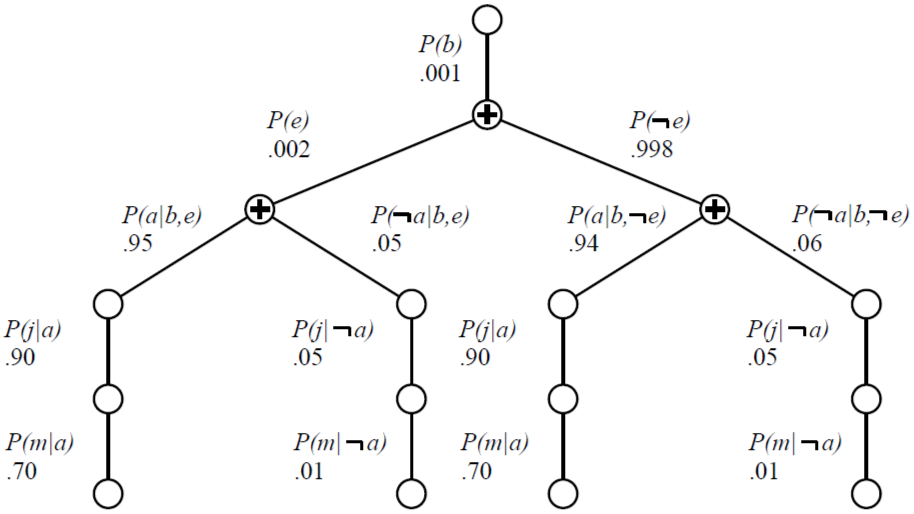
A	P(J|A)
T	.90
F	.05

A	P(M|A)
T	.70
F	.01

- Simple query on the burglary network:
  - $= \alpha\vec{P}(B)\begin{pmatrix} P(e)\sum_a P(a|B,e)P(j|a)P(m|a) \\ +P(\neg e)\sum_a P(a|B,\neg e)P(j|a)P(m|a) \end{pmatrix}$

Evaluation tree

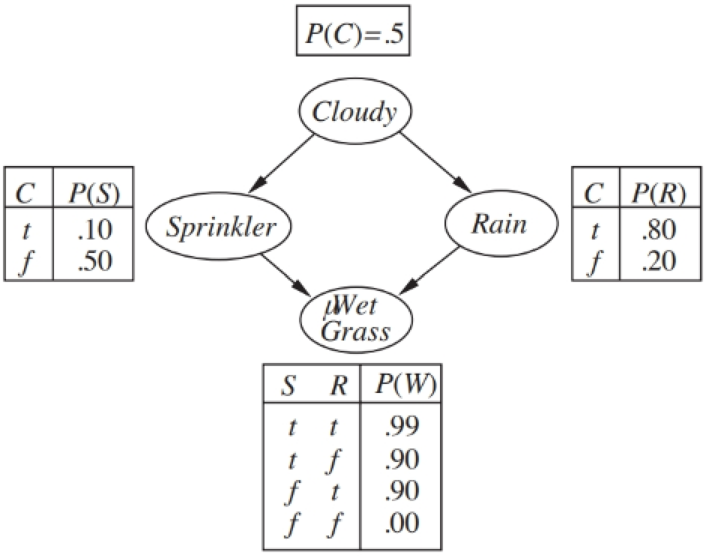
- $\alpha \vec{P}(B) \left(\begin{array}{c} P(e) \left(\begin{array}{c} \vec{P}(a|B,e)P(j|a)P(m|a) \\ +\vec{P}(\neg a|B,e)P(j|\neg a)P(m|\neg a) \end{array} \right) \\ +P(\neg e) \left(\begin{array}{c} \vec{P}(a|B,e)P(j|a)P(m|a) \\ +\vec{P}(\neg a|B,e)P(j|\neg a)P(m|\neg a) \end{array} \right) \end{array} \right)$



Monte Carlo Simulation

- π?

Backyard Network



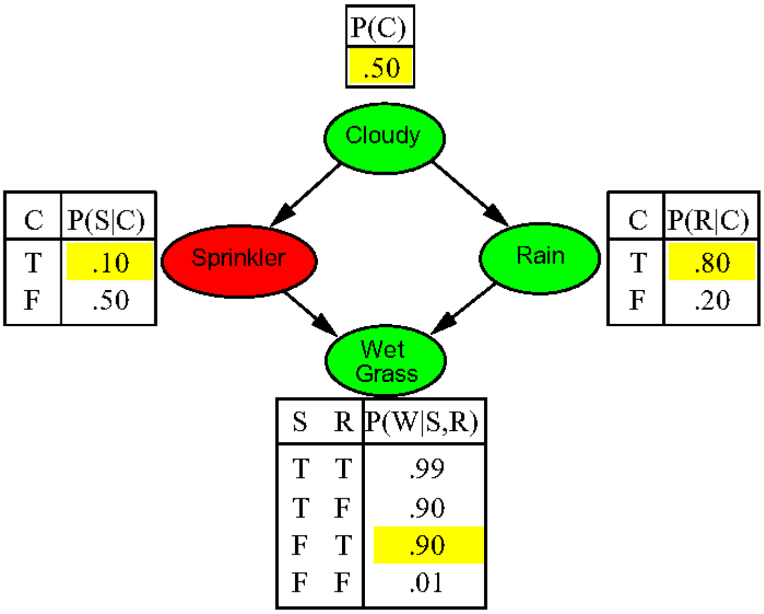
P(C)=.5

C	P(S)
t	.10
f	.50

C	P(R)
t	.80
f	.20

S	R	P(W)
t	t	.99
t	f	.90
f	t	.90
f	f	.00

Prior-Sample algorithm



P(C)
.50

C	P(S|C)
T	.10
F	.50

C	P(R|C)
T	.80
F	.20

S	R	P(W|S,R)
T	T	.99
T	F	.90
F	T	.90
F	F	.01

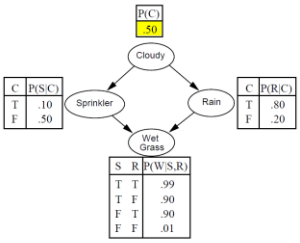
Cloudy	Sprinkler	Rain	WetGrass

.
.
.

Prior-Sample algorithm

- Let $S_{PS}(x_1, \dots, x_n)$ be the probability that a specific event is generated by the sampling.
- $S_{PS}(C = true, S = false, R = true, W = true)$

- Hence, in the limit of large N, we expect 32.4% of the samples to be of this event.



P(C)
.50

C	P(S|C)
T	.10
F	.50

C	P(R|C)
T	.80
F	.20

S	R	P(W|S,R)
T	T	.99
T	F	.90
F	T	.90
F	F	.01

Prior-Sample algorithm

- Let $N_{PS}(x_1, \dots, x_n)$ is the number of times the specific event $x_1, \dots, x_n$

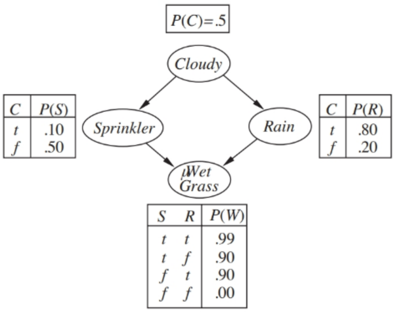
- $P(x_1, \dots, x_n) =$

- $P(x_1, \dots, x_n) \approx$

FFFF	FFFT	FFTF	FFTT	FTFF	FTFT	FTTF	FTTT	TFFF	TFFT	TFTF	TFTT	TTFF	TTFT	TTTF	TTTT	N

Rejection sampling

- Estimate $\vec{P}(Rain|Sprinkler = true)$?
  - Assume $N = 100$, $a + b + e + f = 8$, $c + d + g + h = 19$
  - $\vec{P}(Rain|Sprinkler = true)$
    $=$



FFFF	FFFT	FFTF	FFTT	FTFF	FTFT	FTTF	FTTT	TFFF	TFFT	TFTF	TFTT	TTFF	TTFT	TTTF	TTTT	N
				a	b	c	d					e	f	g	h	100

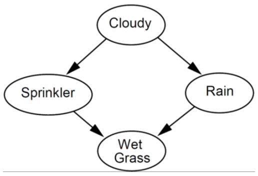
Analysis of rejection sampling

- $\vec{P}(X|e) \approx$
- Problem
  - it rejects so many samples
  - $\vec{P}(Rain|RedSkyAtNight = true)$?
- Solution
  - Weight by probability of evidence given parents
  - Fix the observed variables

FFFF	FFFT	FFTF	FFTT	FTFF	FTFT	FTTF	FTTT	TFFF	TFFT	TFTF	TFTT	TTFF	TTFT	TTTF	TTTT	N
				a	b	c	d					e	f	g	h	100

Markov chain simulation

- Markov chain Monte Carlo (MCMC) algorithm
  - generate each sample by making a random change to the preceding sample
  - In other words, generates a next state by making random changes to the current states
  - Possible states



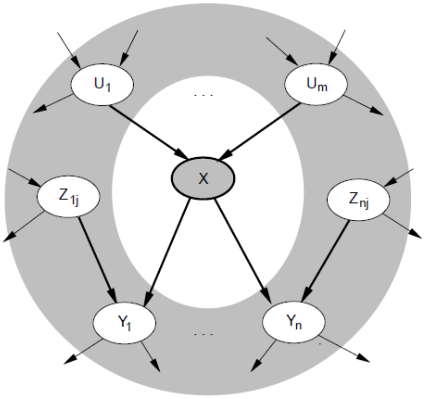
  - Possible states with $Sprinkler = true; WetGrass = true$

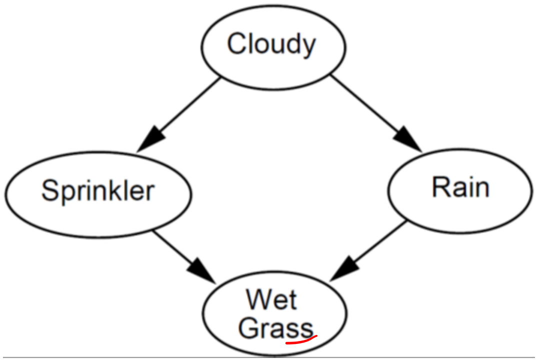
The Markov chain

- With Sprinkler =true;WetGrass=true, there are four states:

Markov blanket

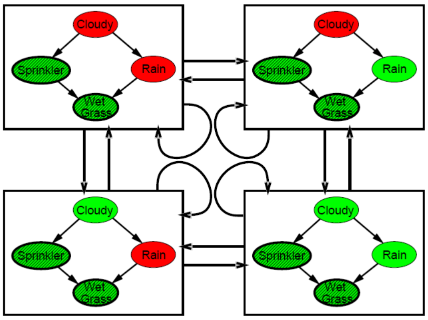
- Markov blanket
  - parents + children + children's parents





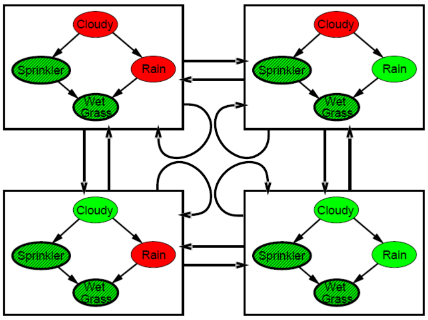
MCMC example

- Estimate $\vec{P}(R|s,w)$
  - An initial state is selected randomly
    - [**true**, true, **false**, true].
  - **Cloudy(randomly)** is chosen, given the current values of its Markov blanket variables
    - sample from



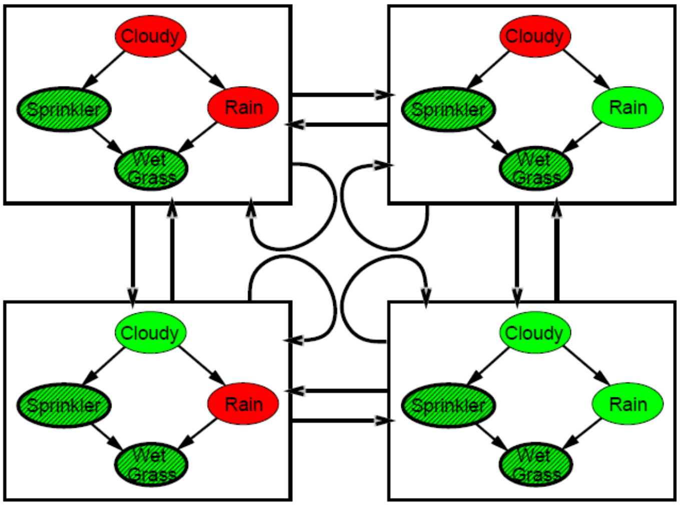
MCMC example

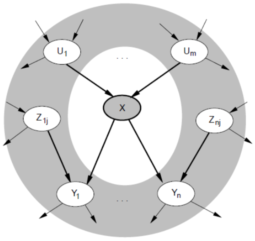
- Estimate $\vec{P}(R|s,w)$
  - Assume $Cloudy = false$ sampled.
    - the current state is
  - **Rain(randomly)** is chosen, given the current values of its Markov blanket variables
    - sample from



The Markov chain

- Estimate $\vec{P}(R|s,w)$

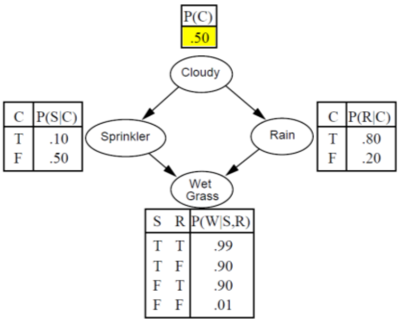




[C,S,R,W]	Count
FTFT	a
FTTT	b
TTFT	c
TTTT	d

Sampling distribution

- $\vec{P}(R|\neg c, s, w)$



P(C)
.50

C	P(S|C)
T	.10
F	.50

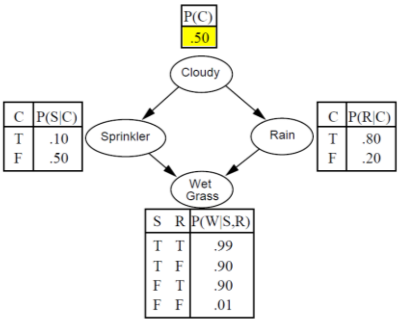
C	P(R|C)
T	.80
F	.20

S	R	P(W|S,R)
T	T	.99
T	F	.90
F	T	.90
F	F	.01

- $\vec{P}(R|c, s, w)$

Sampling distribution

- $\vec{P}(C|s, \neg r)$



P(C)
.50

C	P(S|C)
T	.10
F	.50

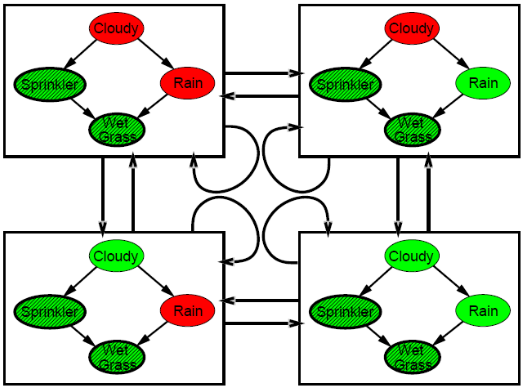
C	P(R|C)
T	.80
F	.20

S	R	P(W|S,R)
T	T	.99
T	F	.90
F	T	.90
F	F	.01

- $\vec{P}(C|s, r)$

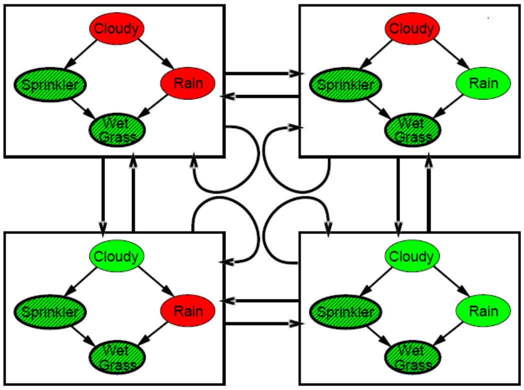
Transition probability

- Let $q(\vec{x} \rightarrow \vec{x}')$ be the probability that the process makes a transition from state $\vec{x}$ to state $\vec{x}'$.
  - $(c, r) \rightarrow (c, r)$
    - Chose C or R:
    - If C is chosen, use
    - Else (R chosen), use
    - $q((c, r) \rightarrow (c, r))$



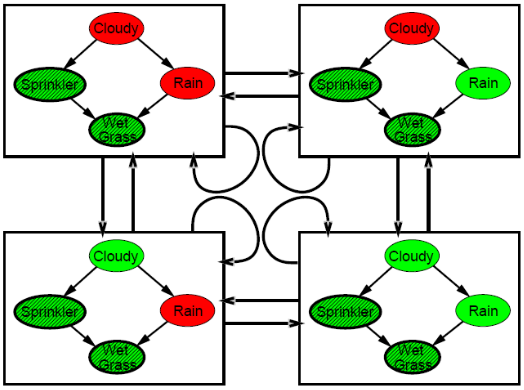
Transition probability

- Let $q(\vec{x} \rightarrow \vec{x}')$ be the probability that the process makes a transition from state $\vec{x}$ to state $\vec{x}'$.
  - $(c, r) \rightarrow (c, \neg r)$
    - Chose C or R:
    - If C is chosen,
    - Else (R chosen),
    - $q((c, r) \rightarrow (c, \neg r))$



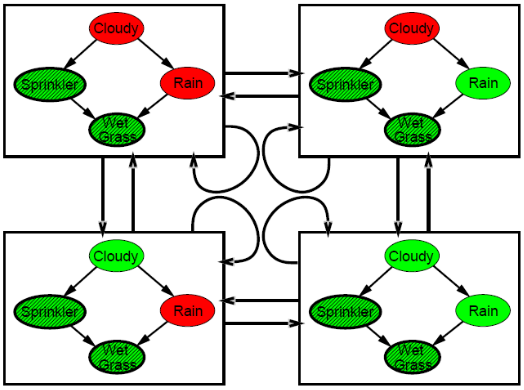
Transition probability

- Let $q(\vec{x} \rightarrow \vec{x}')$ be the probability that the process makes a transition from state $\vec{x}$ to state $\vec{x}'$.
  - $(c, r) \rightarrow (\neg c, r)$
    - Chose C or R:
    - If C is chosen,
    - Else (R chosen),
    - $q((c, r) \rightarrow (c, \neg r))$



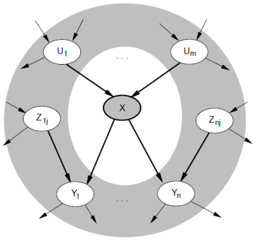
Transition probability

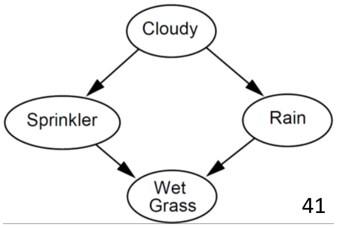
- Let $q(\vec{x} \rightarrow \vec{x}')$ be the probability that the process makes a transition from state $\vec{x}$ to state $\vec{x}'$.
  - $(c, r) \rightarrow (\neg c, \neg r)$
    - Chose C or R:
    - If C is chosen,
    - Else (R chosen),
    - $q((c, r) \rightarrow (c, \neg r))$



Markov blanket

- Should we use $\vec{P}(C|s,r,w)$ instead of $\vec{P}(C|s,r)$?





Quantifying Uncertainty

ARTIFICIAL INTELLIGENCE

JUCHEOL MOON

Uncertainty

- Let action $A_t = \textit{leave}$ for airport t minutes before flight. Will A_t get me there on time?

- Can an agent say, "A_{25} will get me there on time"?
  - partial observability (road state, etc.)
  - uncertainty in action outcomes (flat tire, etc.)

Qualification Problem

- A_{25} will get me there on time
  - if there's no accident on the bridge
  - and it doesn't rain
  - and my tires remain intact
  - etc…
- We are too lazy to enumerate all exceptions, and we do not know all the rules.
- A_{1440} might reasonably be said to get me there on time but I'd have to stay overnight in the airport…

3

3

Probability

- Probability
  - Given the available evidence, A_{25} will get me there on time with probability 0.04
- Probabilistic assertions summarize effects of
  - failure to enumerate exceptions, qualifications, etc.
    -
  - lack of relevant facts, initial conditions, etc.
    -

4

4

Probability

- Bayesian probability
  - Probabilities relate propositions to one's own state of knowledge
    - $P(A_{25} \mid no\ reported\ accidents) = 0.06$
  - Probabilities of propositions change with new evidence
    - $P(A_{25} \mid no\ reported\ accidents, 5\ a.m.) = 0.15$

Making decisions under uncertainty

- Suppose I believe the following:
  - $P(A_{25} \mid \ldots) = 0.04$, $P(A_{90} \mid \ldots) = 0.70$
  - $P(A_{120} \mid \ldots) = 0.95$, $P(A_{1440} \mid \ldots) = 0.9999$
- Which action to choose?
  - Depends on my preferences
    - missing flight vs. airport cuisine, etc.
- Utility theory is used to represent and infer preferences
  - Decision =

Probability basics

- Begin with a set Ω - the sample space
  - w ∈ Ω is a sample point/possible world/atomic event
  - e.g., 6 possible rolls of a die
    - $\Omega =$
- A probability space or probability model is a sample space with an assignment $P(w)$ for every $w \in \Omega$
  -
  -
  -

Probability basics

- An event A is any subset of Ω
  - $P(A) = \sum_{\{w \in A\}} P(w)$
  - e.g., $P(die\ roll < 4) =$

Random variables

- A random variable is a function from sample points to some range
  - e.g., $X = Odd$
    -
- P induces a probability distribution for any r.v. X:
  - $P(X = x_i) = \sum_{\{w:X(w)=x_i\}} P(w)$
  - e.g., $P(Odd = true) =$

9

Propositions

- Think of a proposition as the event (set of sample points) where the proposition is true
- Given Boolean random variables A and B:
  - event a = set of sample points where _______________
  - event $\neg a$ = set of sample points where_______________
  - event $a \land b$ = set of sample points where _______________ and_______________

10

Syntax for propositions

- Propositional or Boolean random variables
  - $Cavity$ (do I have a cavity?)
  - _______________ is a proposition
- Discrete random variables (finite or infinite)
  - $Weather$ is one of
  - _______________ is a proposition
  - Values must be exhaustive and mutually exclusive
- Continuous random variables (bounded or unbounded)
  - $Temp = 61.6$ or $Temp < 62$.

Prior probability

- Prior or unconditional probabilities of propositions
  - $P(Cavity = true) =$
  - $P(Weather = sunny) =$
  - correspond to belief prior to arrival of any (new) evidence
- Probability distribution gives values for all possible assignments:
  - $\vec{P}(Weather) =$

Prior probability

- Joint probability distribution for a set of r.v.s gives the probability of every atomic event on those r.v.s (i.e., every sample point)
  - $\vec{P}(Weather, Cavity)$

Conditional probability

- Conditional or posterior probabilities
  - e.g., $P(cavity|toothache) = 0.8$
  - Whenever toothache is true and we have no further information, conclude that cavity is true with probability 0.8.
- New evidence may be irrelevant, allowing simplification,
  - e.g., $P(cavity|toothache, cancelClass) = P(cavity|toothache) = 0.8$

Conditional probability

- Definition of conditional probability:
  - $P(a|b) =$ if
- Product rule gives an alternative formulation:
  - $P(a \wedge b) =$
- A general version holds for whole distributions,
  - e.g., $\vec{P}(Weather, Cavity) =$

Conditional probability

- What is $\vec{P}(Weather, Cavity)$?

Conditional probability

- What is $\vec{P}(Weather|Cavity)\vec{P}(Cavity)$?

17

17

Conditional probability

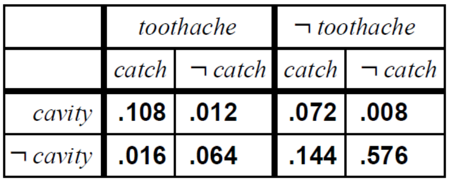
- Chain rule is derived by successive application of product rule:
  - $\vec{P}(X_1, X_2, X_3)$, Let Y = X_1, X_2
  - $\vec{P}(X_1, \ldots, X_n) =$

18

18

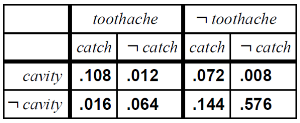
Inference by enumeration

- Start with the joint distribution:
  - P(toothache)
    -
  - P(cavity ∨ toothache)
  -
- For any proposition ϕ, sum the atomic events where it is true:
  - $P(\phi) = \sum_{w:w\models\phi} P(w)$



	toothache		*¬ toothache*	
	catch	*¬ catch*	*catch*	*¬ catch*
cavity	**.108**	**.012**	**.072**	**.008**
¬ cavity	**.016**	**.064**	**.144**	**.576**

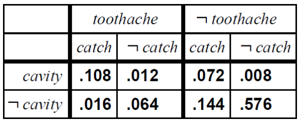
Inference by enumeration



	toothache		*¬ toothache*	
	catch	*¬ catch*	*catch*	*¬ catch*
cavity	**.108**	**.012**	**.072**	**.008**
¬ cavity	**.016**	**.064**	**.144**	**.576**

- Can also compute conditional probabilities:
  - P(¬cavity | toothache)
  - P(cavity | toothache)

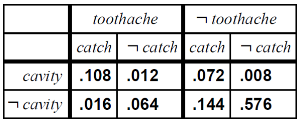
Inference by enumeration



	toothache		*¬ toothache*	
	catch	*¬ catch*	*catch*	*¬ catch*
cavity	**.108**	**.012**	**.072**	**.008**
¬ cavity	**.016**	**.064**	**.144**	**.576**

- $\vec{P}(Cavity\,|\,toothache)$

General inference procedure.

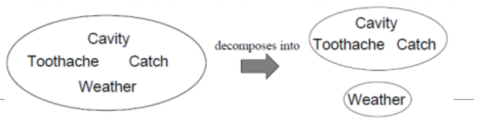


	toothache		*¬ toothache*	
	catch	*¬ catch*	*catch*	*¬ catch*
cavity	**.108**	**.012**	**.072**	**.008**
¬ cavity	**.016**	**.064**	**.144**	**.576**

- $\vec{P}(X|e)$?

  - The query involves a single variable, X
    - Cavity in the example
  - E be the list of evidence variables
    - Toothache in the example
    - e be the observed values for them
  - Y be the remaining unobserved variables
    - Catch in the example

Independence



- A and B are independent iff
  - $P(toothache \wedge catch \wedge cavity \wedge cloudy)$
  - $\vec{P}(Toothache, Catch, Cavity, Weather)$
  - $\vec{P}(A, B) =$
  - $\vec{P}(A|B) =$
  - $\vec{P}(B|A) =$

Bayes' Rule

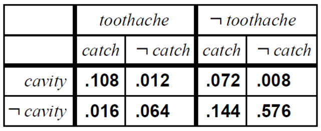
- Product rule $P(a \wedge b) =$
- Bayes’ rule $P(a|b) =$
  - Useful for assessing diagnostic probability from causal probability:
    - $P(cause|effect) =$

Bayes' Rule

- $P(disease|symptoms) =$

- The disease meningitis causes the patient to have a stiff neck, say, 70% of the time, and the prior probability that a patient has meningitis is 1/50,000, and the prior probability that any patient has a stiff neck is 1%.
  - $P(s|m) =$, $P(m) =$, $P(s) =$
  - $P(m|s) =$

Bayes' Rule



	toothache		*¬ toothache*	
	catch	*¬ catch*	*catch*	*¬ catch*
cavity	**.108**	**.012**	**.072**	**.008**
¬ cavity	**.016**	**.064**	**.144**	**.576**

- $\vec{P}(Cavity|ache \wedge catch)$

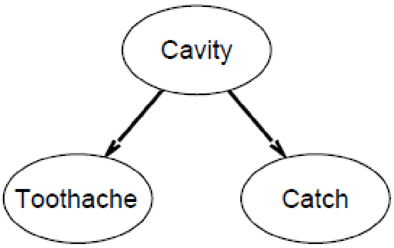
- $\vec{P}(Y|x) =$

Bayes' Rule

- $\vec{P}(Y|x) = \frac{\vec{P}(Y,x)}{P(x)} = \frac{\vec{P}(x|Y)\vec{P}(Y)}{P(x)} = \alpha\vec{P}(x|Y)\vec{P}(Y)$
- $\vec{P}(Y|X) = \alpha\vec{P}(X|Y)\vec{P}(Y)$?

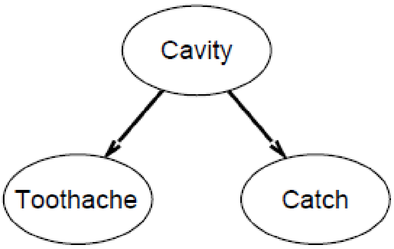
Bayes' Rule and conditional independence

- $\vec{P}(Y|x) = \frac{\vec{P}(Y,x)}{P(x)} = \frac{\vec{P}(x|Y)\vec{P}(Y)}{P(x)} = \alpha\vec{P}(x|Y)\vec{P}(Y)$
- $\vec{P}(Cavity|ache \wedge catch)$



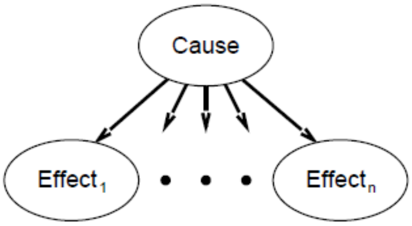
Bayes' Rule and conditional independence

- $\vec{P}(Ache, Catch, Cavity)$



Bayes' Rule and conditional independence

- Naïve Bayes model:
  - $\vec{P}(Cause, Effect_1, \ldots, Effect_n) =$



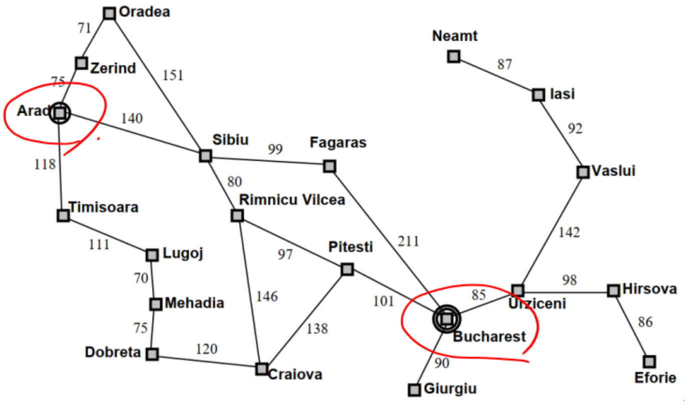
Solving Problems by Searching

ARTIFICIAL INTELLIGENCE

JUCHEOL MOON

Problem-solving agents

- On holiday in Romania; currently in Arad.
- Flight leaves tomorrow from Bucharest



Holiday in Romania

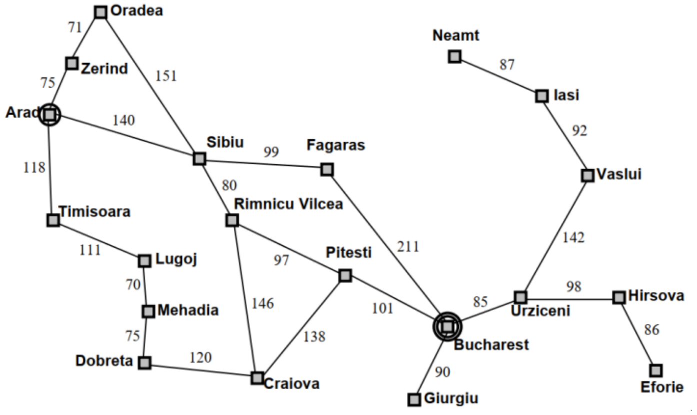
- Formulate goal:

  - be in Bucharest

- Formulate problem:
  - states: Cities
  - actions: drive between cities
- Find solution:

  sequence of cities

Tree search example

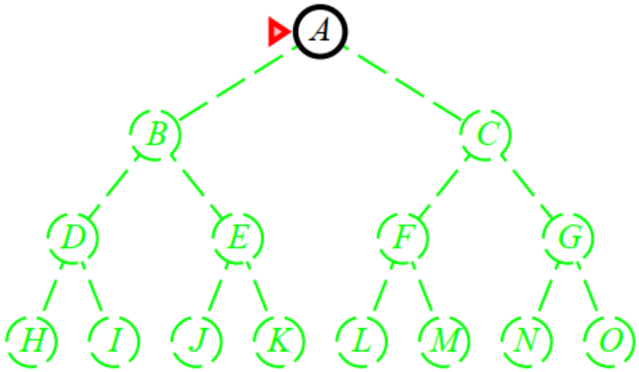


Uninformed search strategies

- Uninformed strategies use only the information available in the problem definition
  - Breadth-first search
  - Depth-first search
  - Uniform-cost search (Dijkstra's algorithm)
  - Depth-limited search
  - Iterative deepening search

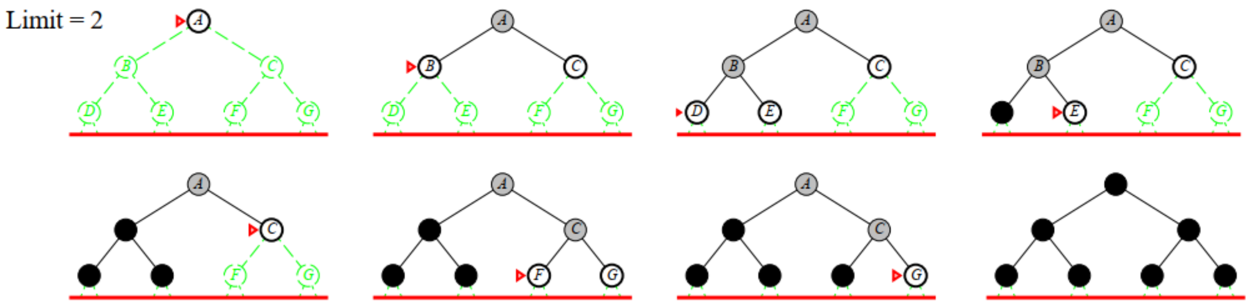
Depth-limited search

- depth-first search with depth limit *l*,
  - i.e., nodes at depth *l* have no successors



Iterative deepening search

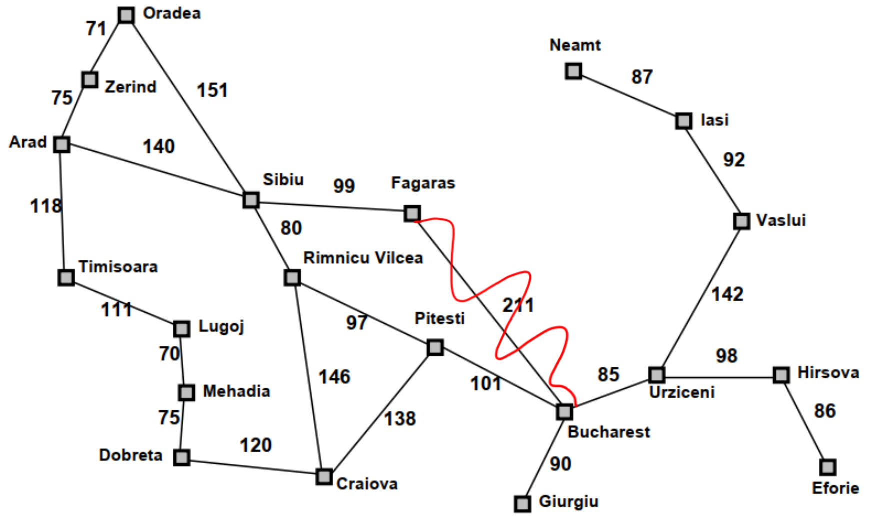
```
function Iterative-Deepening-Search(problem)
    for depth 0 to ∞ do
        depth-Limited-Search(problem, depth)
```



Informed search strategy

- Idea: use an evaluation function for each node
  - estimate of desirability
- Expand most desirable unexpanded node
- Implementation:
  - fringe is a queue sorted in decreasing order of desirability

Romania with step costs in km



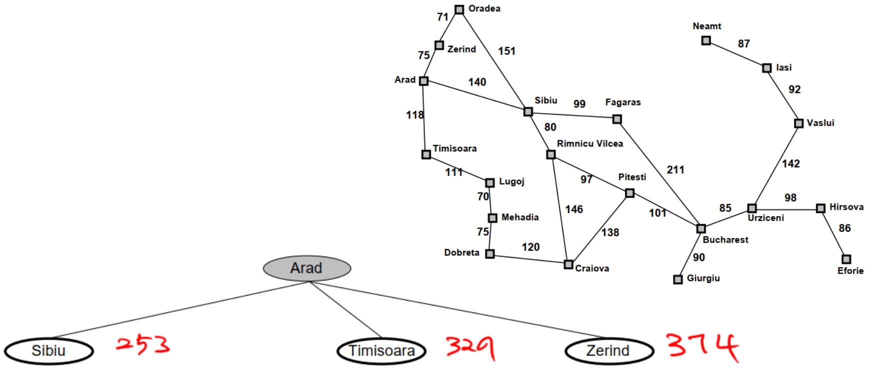
Straight−line distance to Bucharest

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

Greedy search

- Evaluation function $h(n)$ (heuristic)
  - estimate of cost from n to the closest goal
  - $h(n)$ = straight-line distance from n to Bucharest
- Greedy search expands the node that appears to be closest to goal

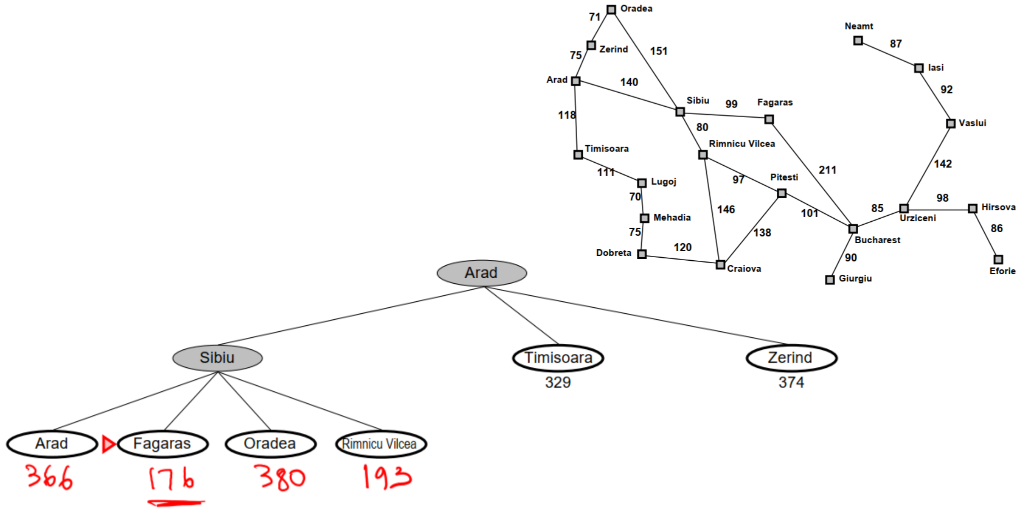
Greedy search example



Straight−line distance to Bucharest

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

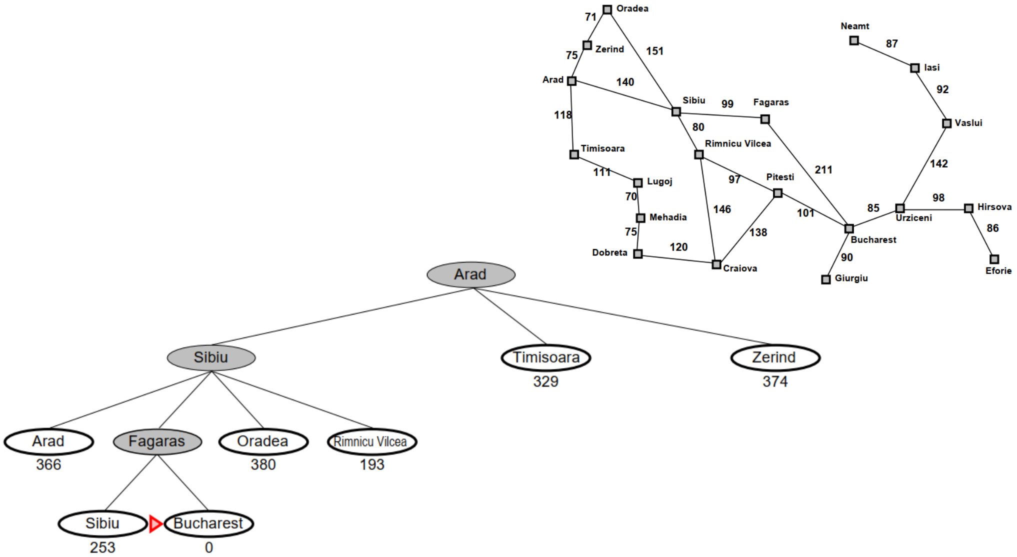
Greedy search example



Straight−line distance to Bucharest

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

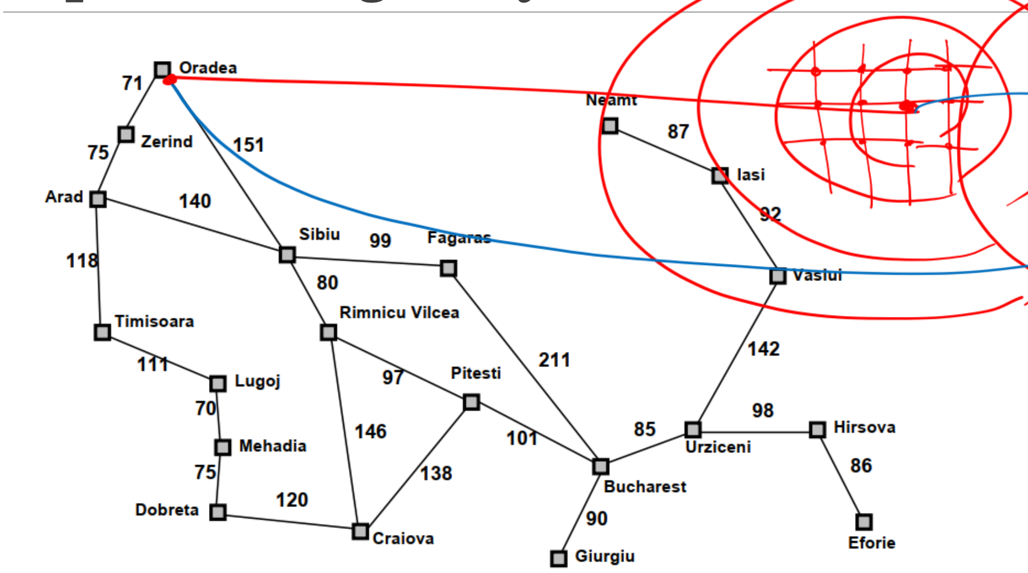
Greedy search example



Straight−line distance to Bucharest

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

Update the greedy search?



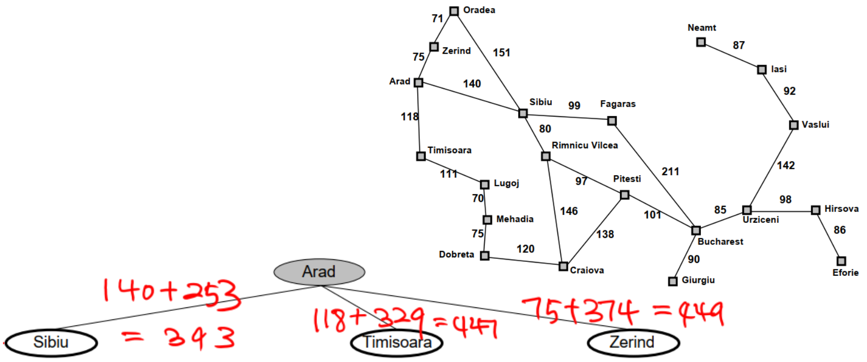
Straight−line distance to Bucharest

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

A∗ search

- Idea: avoid expanding paths that are already expensive
- Evaluation function $f(n) = g(n) + h(n)$
  - $g(n)$ = cost so far to reach n
  - $h(n)$ = estimated cost to goal from n
  - $f(n)$ = estimated total cost of path through n to goal

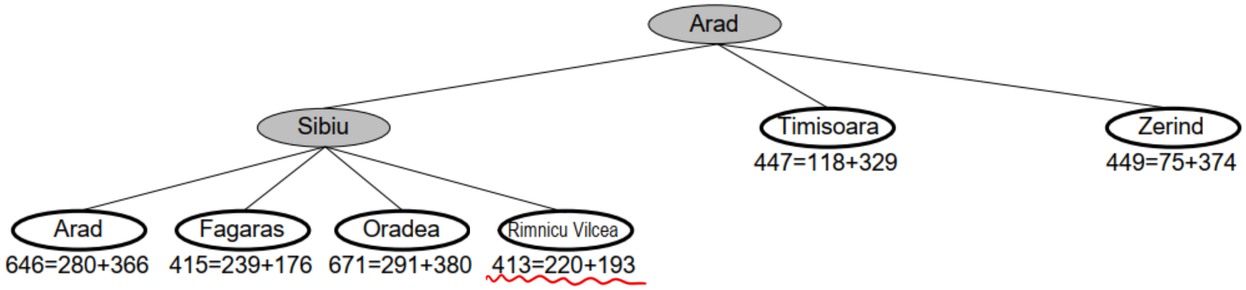
A∗ search example



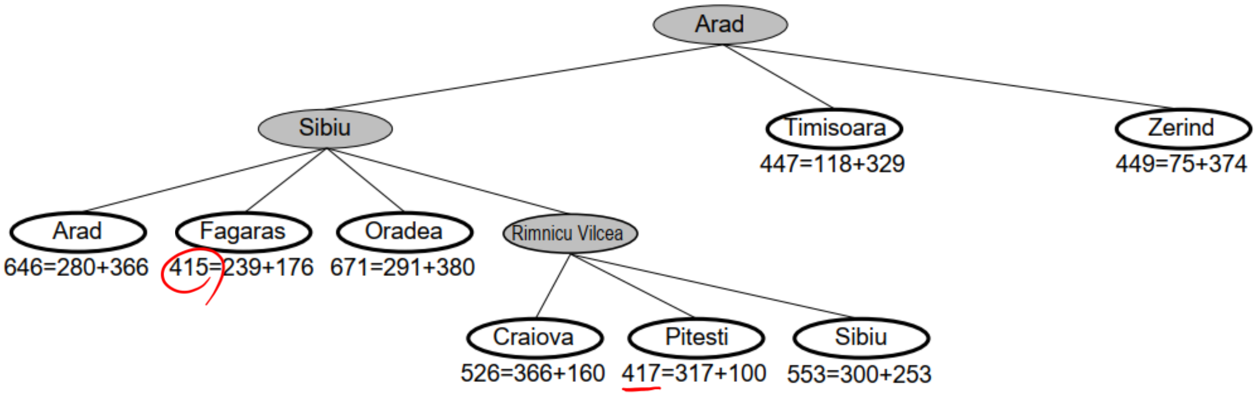
Straight−line distance to Bucharest

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

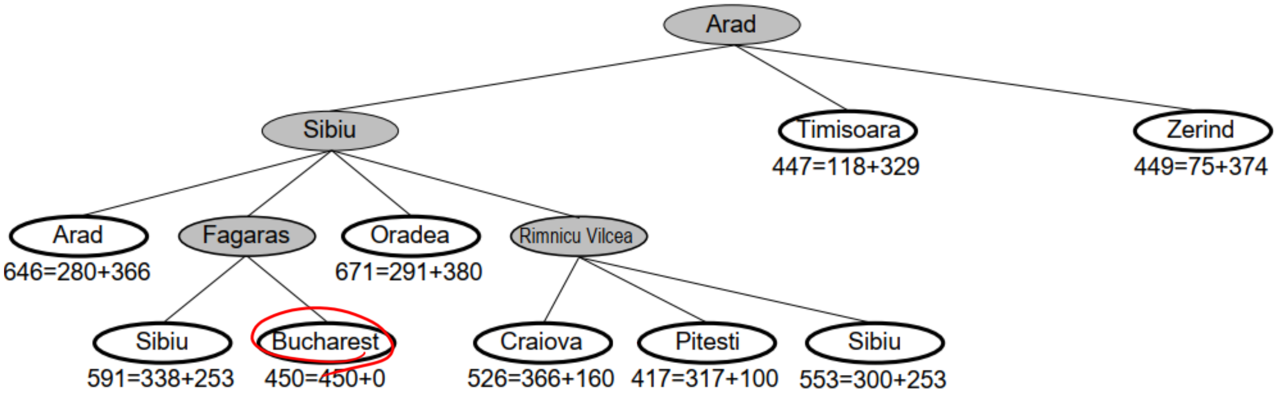
A∗ search example



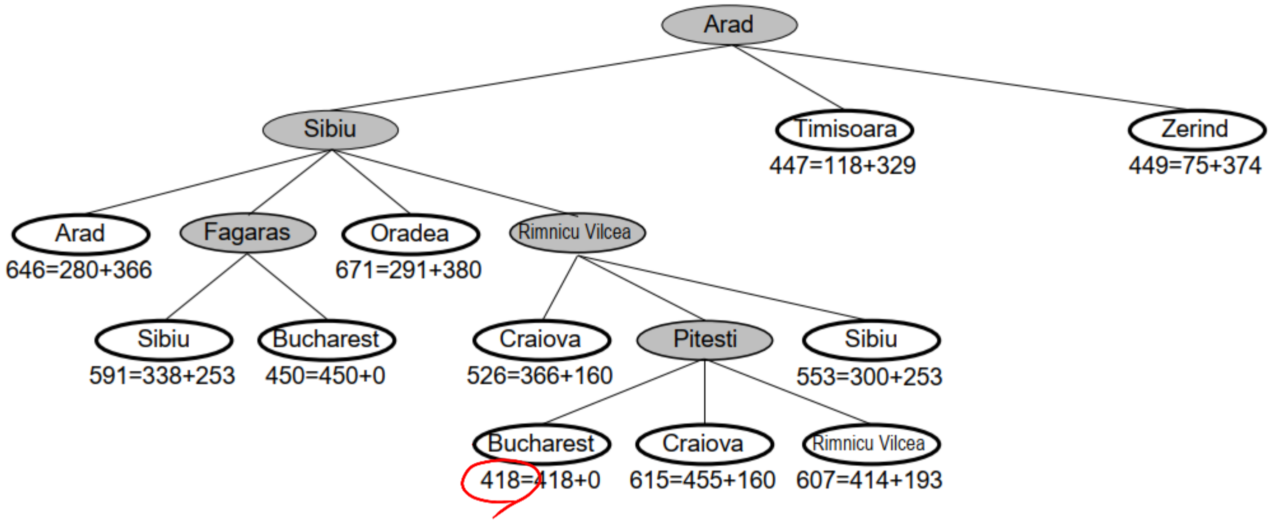
A∗ search example



A∗ search example



A∗ search example



A∗ search

- $h(n)$ = straight-line distance from n to Bucharest
- Can $h(n)$ over estimate the actual road distance?
  - Yes / [✓ No]
- $\underline{h(n) \leq h^*(n)}$ where $h^*(n)$ is the true cost from n?
  - A∗ search uses an admissible heuristic
  - Admissible heuristics are by nature optimistic because they think the cost of solving the problem is ([✓ less] / more) than it actually is.

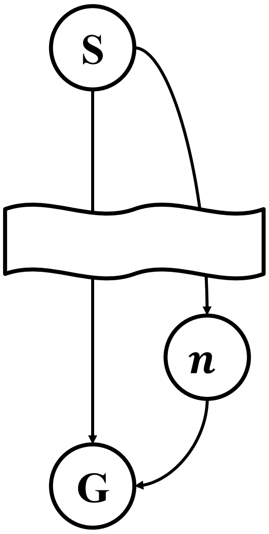
Optimality of A∗

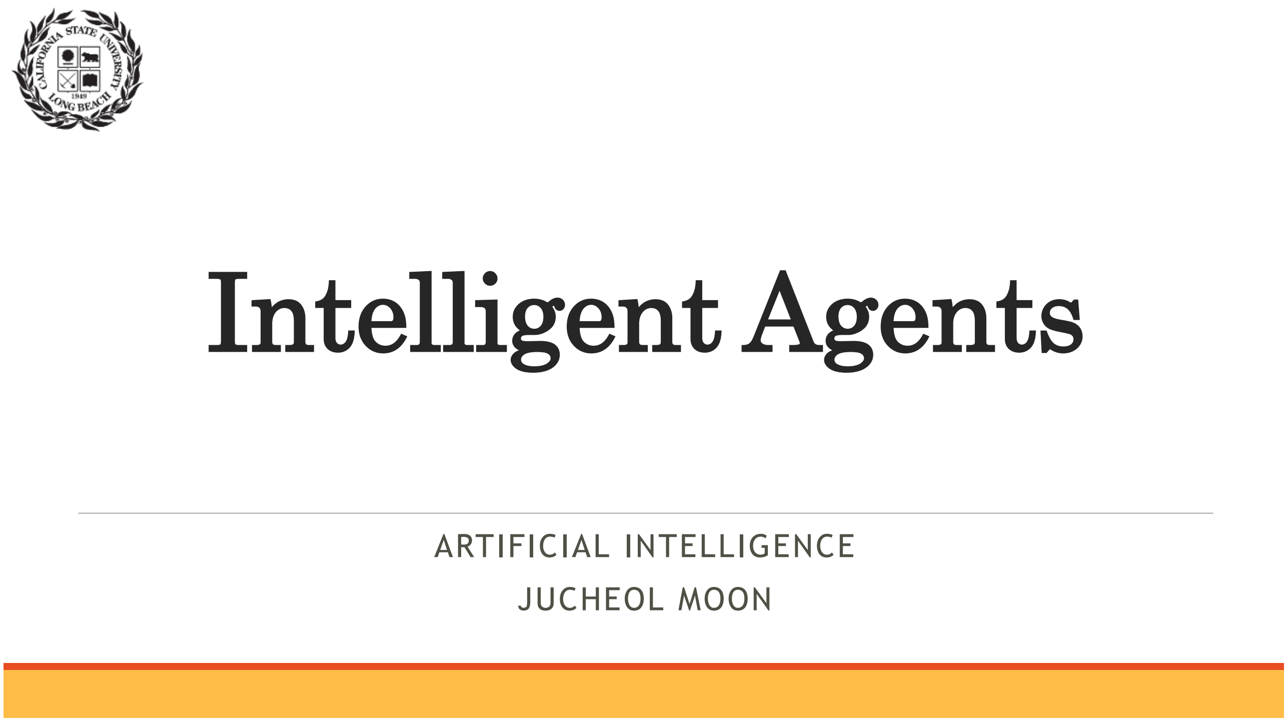
- Assume A* algorithm return a path from S to G, but there is another optimal path.
- If $S \rightarrow n \rightarrow G$ is an optimal

$$f(n) = g(n) + h(n) \leq g(n) + h^*(n) \underline{< g(G)}$$

- However, A* returns the path from S to G

$$f(G) = g(G) + h(G) = g(G)$$
$$\leq g(n) + h(n) \leq g(n) + h^*(n) < g(G)$$



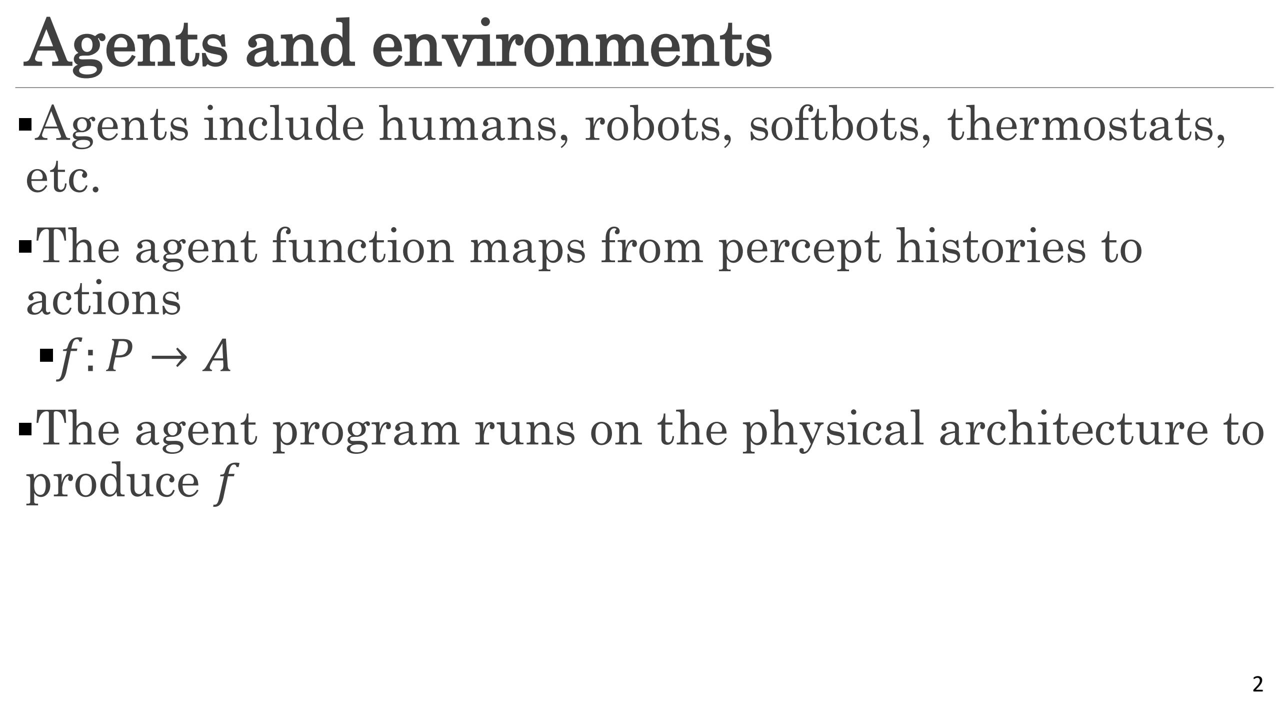


Intelligent Agents

ARTIFICIAL INTELLIGENCE

JUCHEOL MOON

1



Agents and environments

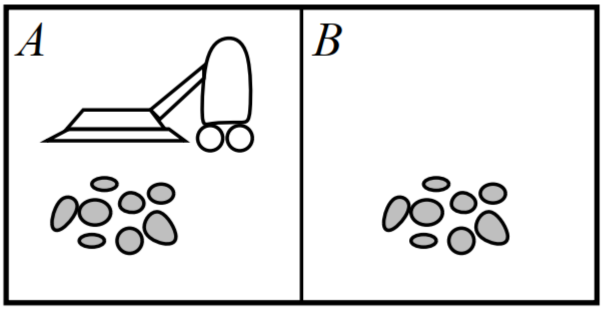
- Agents include humans, robots, softbots, thermostats, etc.
- The agent function maps from percept histories to actions
  - $f: P \rightarrow A$
- The agent program runs on the physical architecture to produce f

2

2

Vacuum-cleaner world

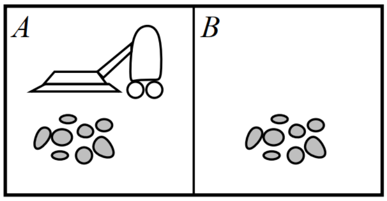
- Equipped component
  - Dirty sensing, move / suck action
- Percepts
  - location and contents
  - e.g., [A; Dirty]
- Actions
  - Left, Right, Suck, NoOp



A vacuum-cleaner agent

- What is the right function?

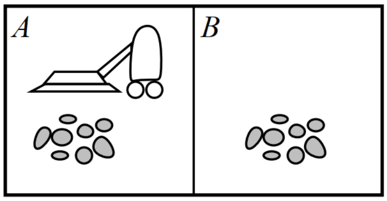
Percept	Action
[A; Dirty]	Suck
[A; Clean]	→
[B; Dirty]	Suck
[B; Clean]	←



A vacuum-cleaner agent

- Can it be implemented in a small agent program?

```
function Vacuum-Agent( [location,status])
   if status = Dirty then return Suck
   else if location = A then return Right
   else if location = B then return Left
```



Rationality

- A rational agent chooses whichever action maximizes the expected value of the performance measure given the percept sequence to date
- Performance measure
  - by the amount of dirt cleaned up?
    - cleaning up the dirt, then dumping it all on the floor, then cleaning it up again, and so on.

Rationality

- Performance measure
  - designing performance measures according to what one actually wants in the environment
- Rational (= or [✓ ≠]) omniscient
  - percepts may not supply all relevant information
- Rational (= or [✓ ≠]) clairvoyant
  - action outcomes may not be as expected
- Hence, rational (= or [✓ ≠]) successful

Task environment

- To design a rational agent, we must specify the task environment
  - Performance measure
  - Environment
  - Actuators
  - Sensors

Task environment of a self-driving car

- Performance measure
  - safety, destination, profits, legality, comfort
- Environment
  - US streets/freeways, traffic, pedestrians, weather
- Actuators
  - steering, accelerator, brake, horn, speaker/display
- Sensors
  - video, accelerometers, gauges, engine sensors, GPS

Environment types

- Fully observable vs. partially observable
  - If an agent's sensors give it access to the complete state of the environment at each point in time, then we say that the task environment is fully observable.
- Single agent vs. multiagent
  - An agent solving a crossword puzzle by itself is clearly in a Single -agent environment, whereas an agent playing chess is in a multi -agent environment.

Environment types

- Deterministic vs. stochastic
  - If the next state of the environment is completely determined by the current state and the action executed by the agent, then we say the environment is deterministic.
- Episodic vs. sequential
  - In an episodic task environment, the agent's experience is divided into atomic episodes. The next episode does not depend on the actions taken in previous episodes.

Environment types

- Static vs. dynamic
  - If the environment can change while an agent is deliberating, then we say the environment is dynamic for that agent.
- Discrete vs. continuous
  - The chess environment has a finite number of distinct states (excluding the clock). Chess also has a discrete set of percepts and actions.

Environment types of a self-driving car

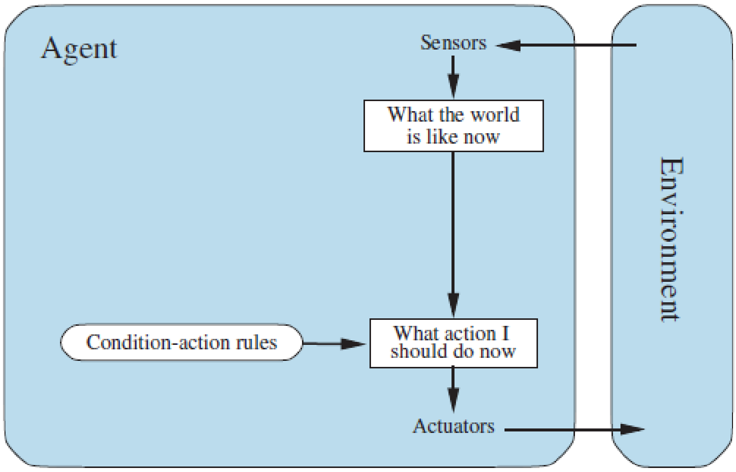
- Observable?
  - Partially
- Agents?
  - Multi
- Deterministic?
  - Stochastic
- Episodic?
  - Sequential
- Static?
  - Dynamic
- Discrete?
  - Continuous

Agent types

- Four basic types in order of increasing generality:
  - Simple reflex agents
  - Model-based reflex agents
  - Goal-based agents
  - Utility-based agents
  - Learning agents

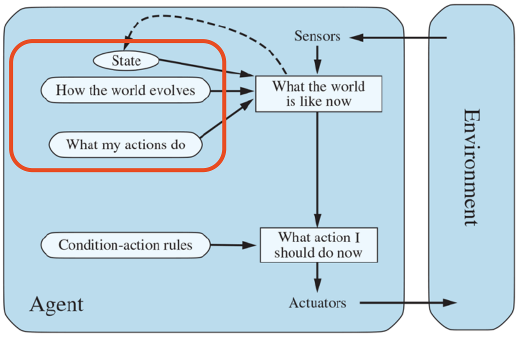
Simple reflex agents

- if car-in-front-is-braking
  - then initiate-braking.



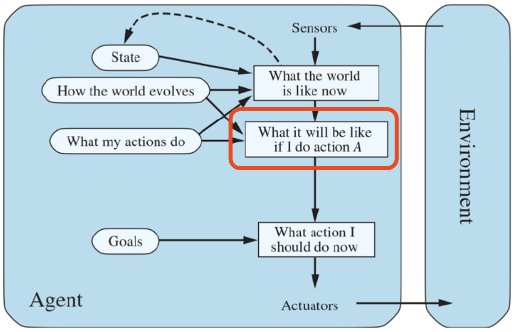
Model-based reflex agents

- We need some information about how the agent’s own actions affect the world
  - When the agent turns the steering wheel clockwise
    - The car turns to the right
  - After driving for five minutes northbound on the freeway
    - One is usually about five miles north of where one was five minutes ago.



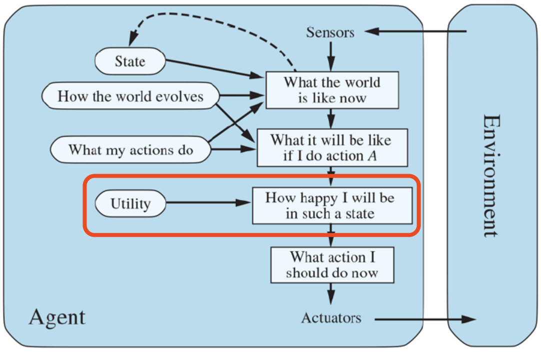
Goal-based agents

- The agent program can combine this with the model to choose actions that achieve the goal
  - At a cross road, an self-driving car can turn left, turn right, or go straight on.
  - The correct decision depends on where the taxi is trying to get to.


Sensors
State
How the world evolves
What my actions do
What the world is like now
What it will be like if I do action A
Goals
What action I should do now
Agent
Actuators
Environment

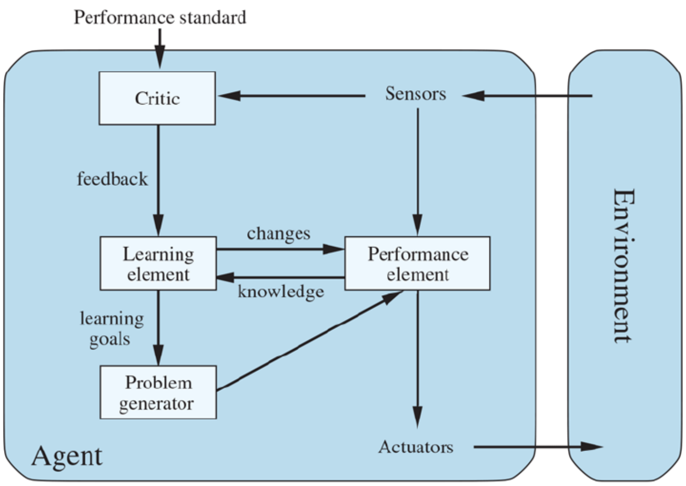
Utility-based agents

- Goals alone are not enough to generate high-quality behavior in most environments
  - Goal: get the taxi to its destination
  - Plus
    - Quicker, safer, more reliable, or cheaper


Sensors
State
How the world evolves
What my actions do
What the world is like now
What it will be like if I do action A
Utility
How happy I will be in such a state
What action I should do now
Agent
Actuators
Environment

Learning agents

[redacted]



Autonomous agents

- They can perceive its environment, make decisions, and take actions toward achieving goals **without human intervention**.
  - Autonomous Meeting Assistants (Fireflies.ai, Otter.ai)
    - Join meetings, record/transcribe, summarize action items, and schedule tasks in project management tools.
  - Autonomous Trading Bots
    - Analyze stock/crypto market data, news, and trends → make buy/sell decisions under predefined risk constraints.
  - Autonomous Content Curators
    - Continuously scan blogs, social media, or forums → summarize trends → auto-publish digests (e.g., “daily AI research highlights”).