INF5390 – Artificial Intelligence

AI – Introduction and Overview

Roar Fjellheim

INF5390 – Artificial Intelligence (AI)

- Purpose of the course
  - Present «state-of-the-art» of AI with *Intelligent Agents* as a common conceptual model
  - Provide sufficient background for students so that they may pursue own reading of AI literature and initiate own AI projects

- The first lecture
  - High-level view of history and state of AI
  - Prepare for the rest of the course

Text book

- **Artificial Intelligence - A Modern Approach** (AIMA), Third edition
  - Stuart Russel, Peter Norvig
  - Prentice Hall, 2010
  - “The Intelligent Agent Book”

- Lectures will mainly be based on textbook contents
- Exercises will be given based on textbook material

Course contents

1. AI – Introduction and Overview
2. Intelligent Agents
3. Solving Problems by Searching
4. Logical Agents
5. First-Order Logic
6. Knowledge Engineering in FOL
7. Knowledge Representation
8. Agents That Plan
9. Planning and Acting
10. Agents That Reason Under Uncertainty
11. Making Simple Decisions
12. Agents That Learn
13. Reinforcement Learning
14. Agents That Communicate
15. Foundations and Prospects

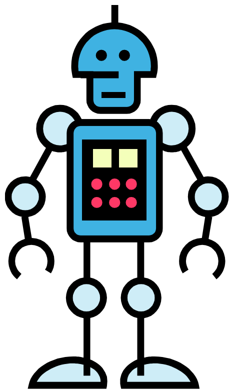
Course format

- Lectures at UiO/IfI – see lecture plan
- Lecture notes published at the course site (pdf)
  - http://www.uio.no/studier/emner/matnat/ifi/INF5390/
- Two written exercises to be delivered during the semester
- Written 4-hour examination
- Lecturer
  - Roar Fjellheim, Adjunct professor (Prof. II)
  - Computas AS
  - E-mail raf@computas.no
  - Tel. 901 25 705

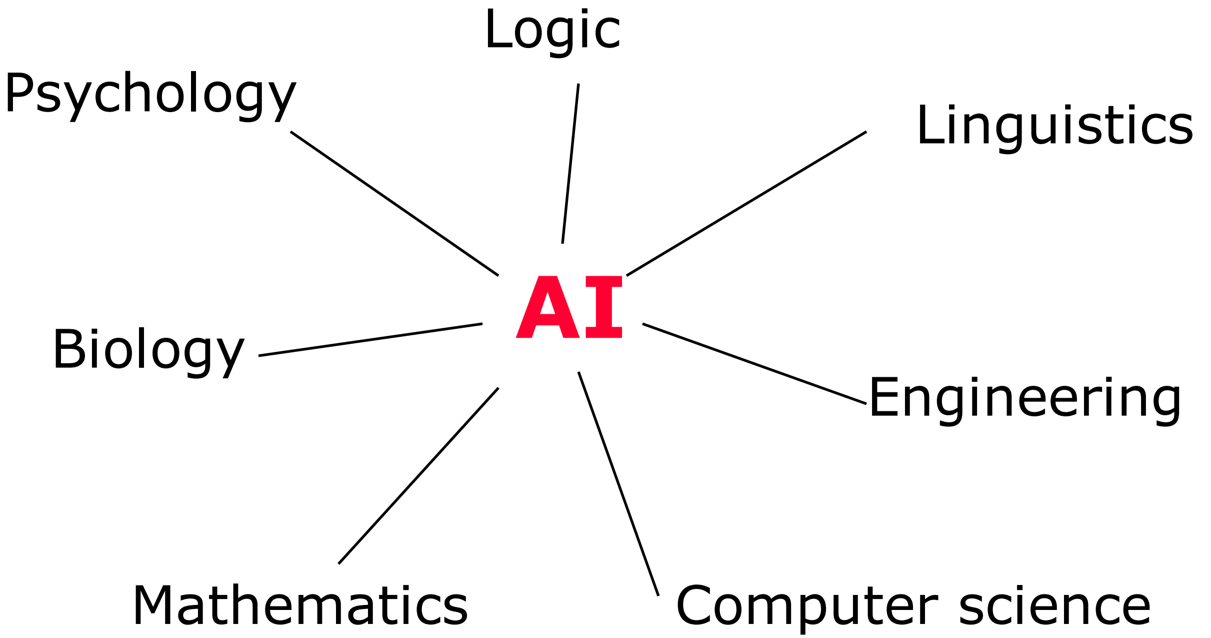
What is AI?

- Cross-disciplinary profession with dual goals:
  - √ Improve our understanding of human intelligence
  - √ Enable us to build intelligent systems
- Common methodology:
  - √ Construct computational models of intelligence

Systems thinking like humans	Systems thinking rationally
Systems acting like humans	**Systems acting rationally**



AI – Eclectic and cross disciplinary



Brief history of AI

- AI (Artificial Intelligence) formed (1945 - 1960)
  - Early connectionism
  - Dartmouth workshop (1956) – John McCarthy
- Enthusiasm, great expectations (1960 - 1970)
  - GPS - General Problem Solver – A. Newell and H. Simon
  - Lisp
- Realism takes hold (1965 - 1975)
  - Combinatorial explosion
  - Natural language systems fail

Brief history (cont.)

- Knowledge based systems (1970 - 1980)
  - "Knowledge is power"
  - Expert systems– Edward Feigenbaum
- AI is industrialized (1980 - 1990)
  - Startup companies
  - AI products
- New connectionism (1985 -)
  - Neural networks for learning
  - Symbol processing vs. connectionism

Brief history (cont.)

- AI becomes a science (1985 -)
  - From ad hoc to structured
  - Scientific methods and ”accumulation”
- Intelligent agents (1990 -)
  - Distributed AI and Internet/Web
  - Common frameworks/conceptualizations
- AI applications – ”second wave” (1995 -)
  - Embedded applications and integration
  - “Invisible AI”?
- AI on the Web (2005 -)
  - Semantic web – Tim Berners-Lee
  - AI for intelligent search

AI today – Myth or reality?

- "Today's AI does not try to re-create the brain. Instead, it uses machine learning, massive data sets, sophisticated sensors, and clever algorithms to master discrete tasks."

  Wired, Jan. 2011

Watson won!

Watson wins 'Jeopardy!' finale

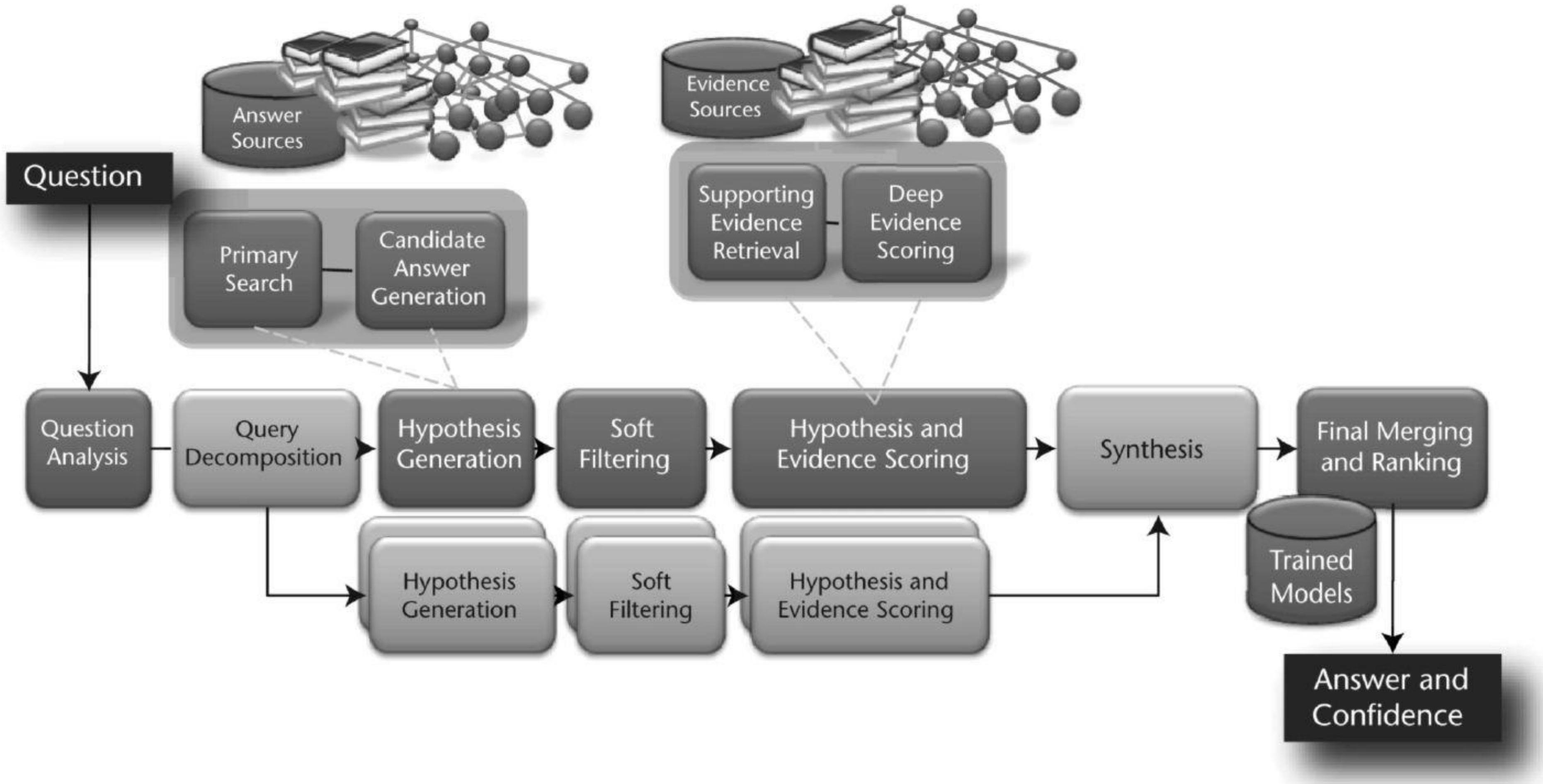
February 16, 2011

"Jeopardy!" has a new champion, and its name is Watson. During the Wednesday finale of the three-day "Jeopardy!" challenge that pitted all-stars Ken Jennings and Brad Rutter against an IBM-engineered supercomputer, the machine ultimately beat the men.

Watson design principles

- *Massive parallelism*
  - √ Exploit massive parallelism in the consideration of multiple interpretations and hypotheses.
- *Many experts*
  - √ Facilitate the integration, application, and contextual evaluation of a wide range of loosely coupled probabilistic question and content analytics.
- *Pervasive confidence estimation*
  - √ No component commits to an answer; all components produce features and associated confidences, scoring different question and content interpretations. An underlying confidence-processing substrate learns how to stack and combine the scores.
- *Integrate shallow and deep knowledge*
  - √ Balance the use of strict semantics and shallow semantics, leveraging many loosely formed ontologies.

Watson high-level architecture

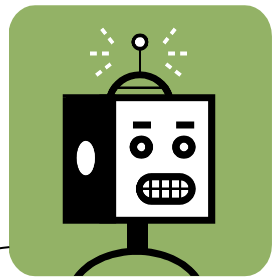


Other examples of real world AI

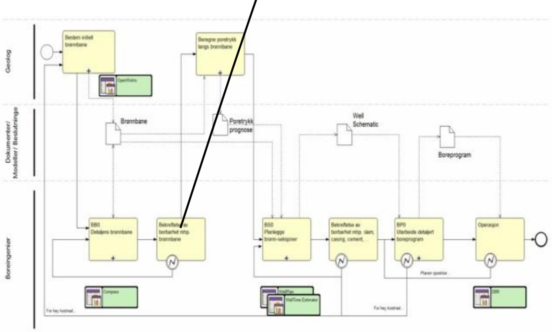
- Warehouses with mobile robots that store and fetch items upon request
- Agents following stock exchange trends, outperforming humans in selling and buying
- Automated scanning and interpretation of X-ray images in medicine
- Image matching and recognition in mobile Internet search (Google Goggles)
- Neural networks for fraud detection in banking transactions
- Autonomous cars navigating and driving unassisted in normal urban traffic (still in trial mode …)

AI in oil drilling – AKSIO (Statoil/Computas)

Reason about what information is needed for an activity, and go get it!



Information resources (people, databases, tools, ..)



Process activities

	People/Networks							Documents/databases													IT Tools							
Knowledge source / *Workstep*	GFC boreteam	GFA/B boreteam	Andre Tampen boreteam	Leverandører (SLB, HLB)	Fagledere	Fagnettverk	Subsurface Support Center	Rig Action Plan (RAP)	Risiko dokument for brønnen	Arena / Team Site	Dok. tidligere brønner	Egne sluttrapporter	Leverandørers sluttrapporter	DBR	**DBR Experience**	MDP Gullfaks	Annen styrende dok.	Beste Praksis	lokal Beste Praksis (GF)	Leverandørers Web-kataloger	OpenWorks	Compass	WellPlan	Welltime Estimator	Office (doc, pdf, ppt, xls)	Stress Check (Casing Design)	OptiCem (leverandør simulering)	Virtual Hydraulics (lev. sim.)
Detail well path																												
Check dogleg	x	a		a	a						a				a			a	a		x	x						
Check interference with previous wells	x	a		a	a						a				a			a	a		x	x						
Overall method choice	x	x	x	x	x	x	x				x	x	x	x	x	x	x	x	x	x			x		x	x	x	x
Confirmation of ability to drill the well path											a											x	x					
Plan well sections																												
Trial with standard hole and casing dimensions	x			a	a						a				a			x	x				x			x	x	x
Decide depth of foringsrørsko and nummber of sections	x			a	a				x	x	a				a	x	x	x	x				x			x	x	x
Perform simulation ECD and T&D	x	a		a	a						a				a								x				x	x
Decide further progression planning	x	a	a	a	a						a									a					x			
With small margins contact SSC / Experts	x	a	a	a	a	a	x																					
Updated estimate time/costs for well drilling	x													x	a				a					x	x			
Confirmation of ability to drill in relation to slam, casing, cement	x			a	a														a				x		x	x	x	x
Develop detailed drillng program																												
Document detailed method choice	x			a					x	x		x	x		a				a						x			
Document chosen drilling technology	x			a					x	x		x	x		a				a						x			
Chose supplier / undersupplier	x	a	a	x	x	a	a			x	a	x			a			a	a	x					x			
Describe special well HSE conditions	x	a	a	a	a	a	a		x	x	a	x	x		a	a	a	a	a						x			
Choose starting point for doc. (maximize reuse)											x														x			
Work out detailed timeplan in DBR	x			a										x											x			
Detail detailed operation plan	x	a	a	a	a	a	a	x		x	a			x	a			a	a	a					x			
Describe detailed completion plan	x	a	a	a	a	a	a	x		x	a			x	a			a	a	a					x			

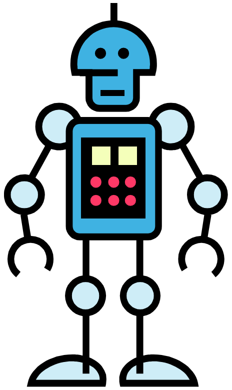
Legend: x = used, a = can also be used

“Clever AI algorithms” – An inventory

- Intelligent agents
- Search algorithms
- Constraint satisfaction
- Logic and reasoning
- Knowledge representation
- Planning systems
- Uncertain information
- Making decisions
- Machine learning
- Neural networks
- Genetic algorithms
- Semantic web

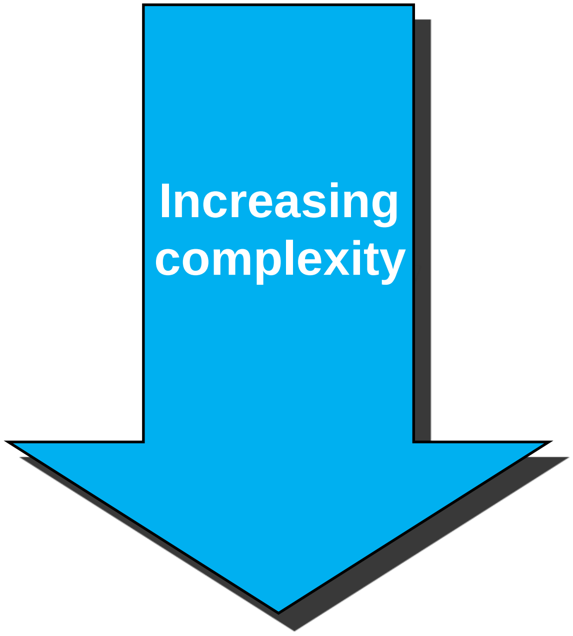
Agents, rationality and AI

- An *agent* is anything that can be viewed as *perceiving* its environment through *sensors* and *acting* upon that *environment* through *actuators*
- A *rational agent* is an agent that for each situation selects the *action* that maximizes its *performance* based on its *perception* and built-in *knowledge*
- **The task of AI is to build problem solving computer programs as rational agents**

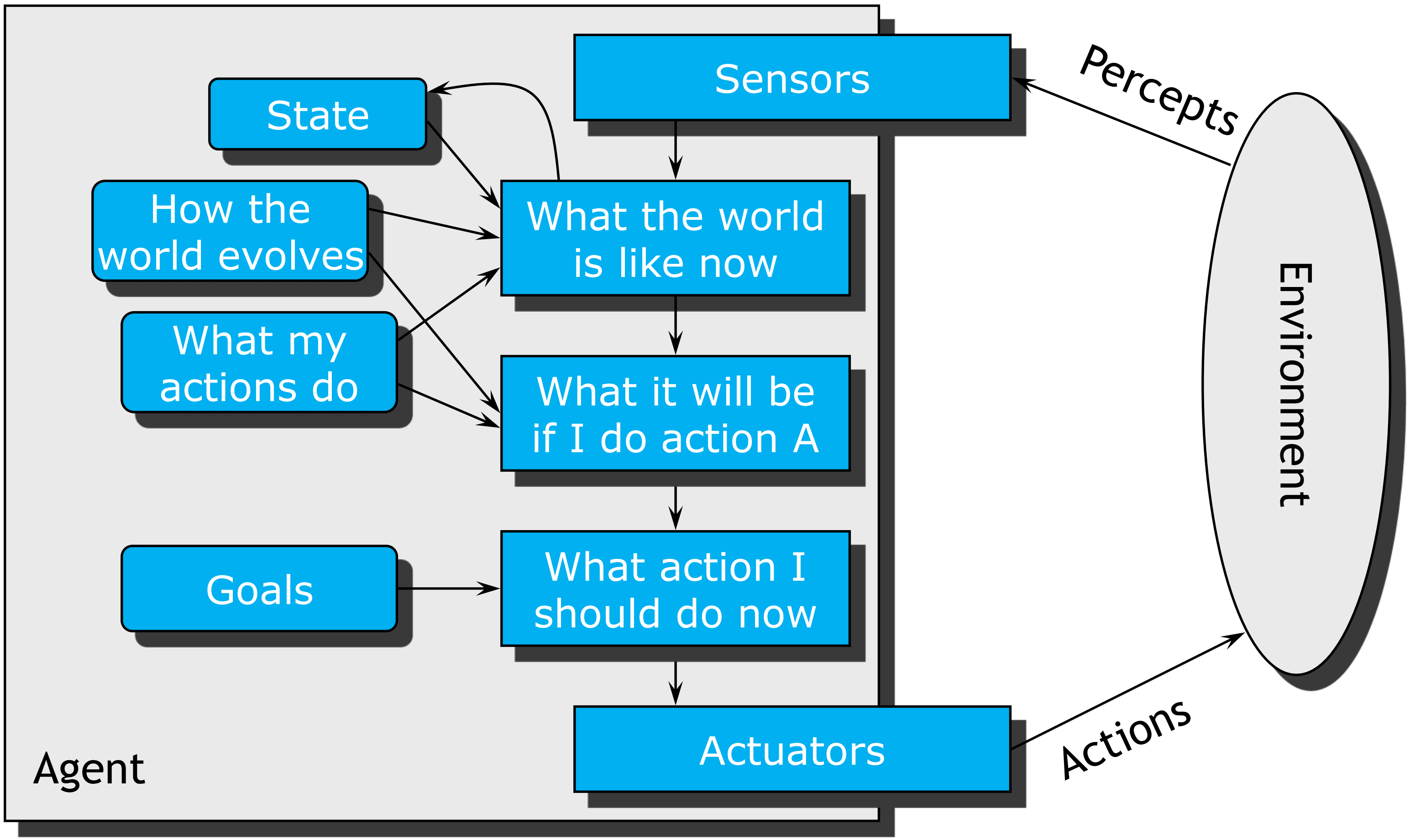


Agent categories

- Table driven agent
- Simple reflex agent
- Model-based reflex agent
- Goal-based agent
- Utility-based agent
- Learning agents

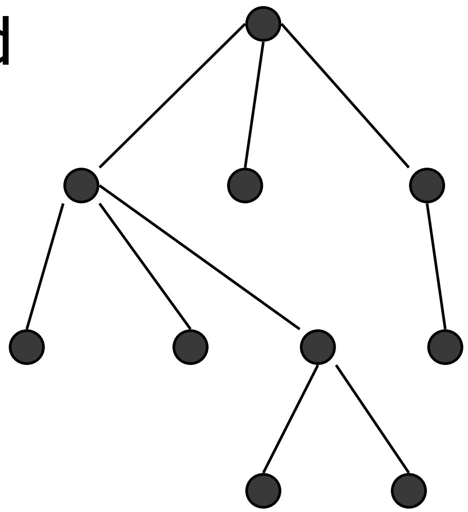


Example: Goal-based agent



Solving problems by searching

- The search starts in an *initial state*
- Then, it iteratively explores the state space by selecting a state node and applying operators to generate *successor nodes* until it finds the *goal state* node or has to give up
- The choice of which node to expand at each level is determined by the *search strategy*
- The part of the state space that is explored is called t
he *search tree*



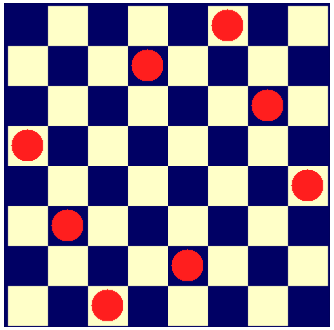
Comparing uninformed search strategies

Criterion	**Breadth-first**	**Uniform-cost**	**Depth-first**	**Depth-limited**	**Iterative deepening**	**Bi-directional**
Complete	Yes	Yes	No	No	Yes	Yes
Time	b^d	$b^{1+c/e}$	b^m	b^l	b^d	$b^{d/2}$
Space	b^d	$b^{1+c/e}$	bm	bl	bd	$b^{d/2}$
Optimal	Yes	Yes	No	No	Yes	Yes

b - branching factor m - maximum depth of tree
d - depth of solution l - depth limit
c – cost of solution e – cost of action

Constraint satisfaction problem

- A constraint satisfaction problem (CSP) is a search problem defined by
  - √ *Variables* - define search states
  - √ *Constraints* - included in the goal test



- Example: 8 queen problem
  - √ *Variables* - location of each queen on the board
  - √ *Constraints* - no queen in same row, column or diagonal
- Many *design* and *scheduling* problems can be formulated as CSP

CSP search strategies

- CSP can be solved by general search methods, but special algorithms are more efficient
- General flow in CSP search
  - *Assign* variable values from their domains
  - *Backtrack* if a constraint is violated by the assignment
  - *Propagate* constraints reducing variable domains
  - *Backtrack* if any domain becomes empty
- Efficient CSP search algorithms are required for many important AI applications and are available as commercial packages

Logic for representation and reasoning

- First-order logic is based on "common sense" concepts:
  - *Objects*: people, houses, numbers, ..
  - *Properties*: tall, red, ..
  - *Relations*: brother, bigger than, ..
  - *Functions*: father of, roof of, ..
  - *Variables*: x, y, .. (takes objects as values)

$$\forall m,c\ Mother(c) = m \Leftrightarrow Female(m) \wedge Parent(m,c)$$

$$\forall w,h\ Husband(h,w) \Leftrightarrow Male(h) \wedge Spouse(h,w)$$

$$\forall g,c\ Grandparent(g,c) \Leftrightarrow \exists p\ Parent(g,p) \wedge Parent(p,c)$$

$$\forall x,y\ Sibling(x,y) \Leftrightarrow x \neq y \wedge \exists p\ Parent(p,x) \wedge Parent(p,y)$$

- First-order logic is the most important and best understood logic in philosophy, mathematics, and AI

Commitments of some logic languages

Language	**Ontological Commitment**	**Epistemological Commitment**
Propositional logic	Facts	True/false/unknown
First-order logic	Facts, objects, relations	True/false/unknown
Temporal logic	Facts, objects, relations, times	True/false/unknown
Probability theory	Facts	Degree of belief 0 .. 1
Fuzzy logic	Facts w/degree of truth	Known interval value

Inference rules and chaining

- An *inference rule* is a standard pattern of inference that can be applied to drive chains of conclusions leading to goal

$$\frac{\alpha \Rightarrow \beta, \quad \alpha}{\beta}$$

- Forward chaining
  - Start with sentences in KB and generate consequences
  - Uses inference rules in forward direction
  - Also called *data-driven* procedure
- Backward chaining
  - Start with goal to be proven and look for premises
  - Uses inference rules in backward direction
  - Also called *goal-directed* procedure

Agents and knowledge bases

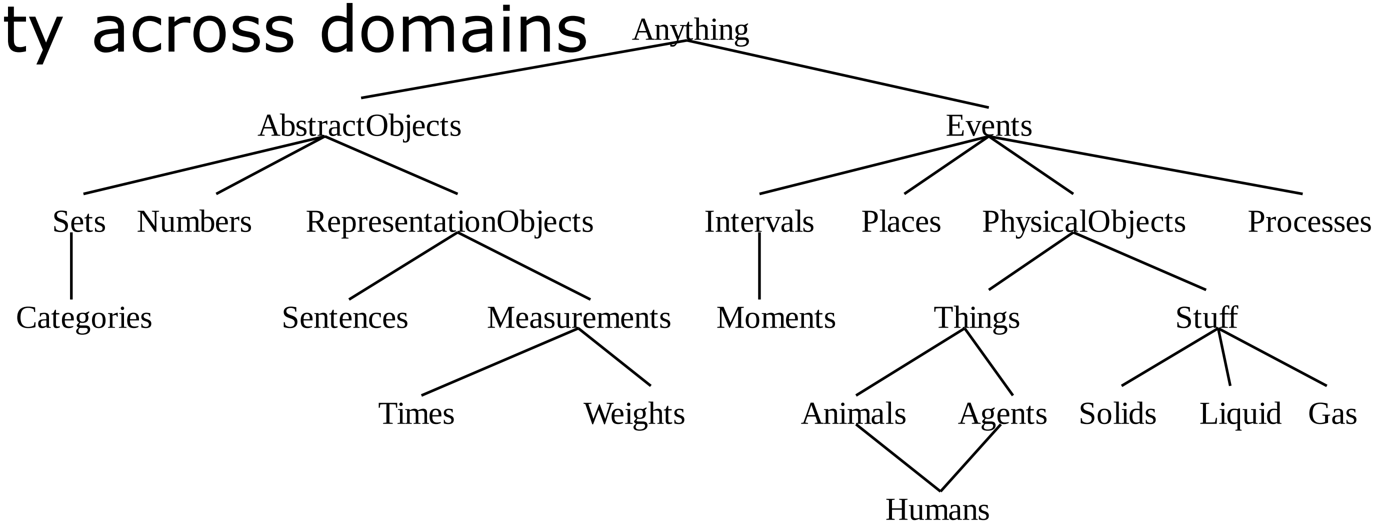
- A *knowledge base* is a set of representations of facts about the world, called *sentences*, expressed in a *knowledge representation language*
- Knowledge base (KB) interface
    - TELL(*KB*, *fact*) - Add a new fact
    - *fact* <= ASK(*KB*, *query*) - Retrieves a fact
    - RETRACT(*KB*, *fact*) - Removes a fact
- A knowledge-base agent can be built by TELLing it what it needs to know (*declarative* approach)
- The agent can be used to solve problems by ASKing questions

Knowledge engineering

- Knowledge engineering is the process of building a knowledge base for a domain
  - √ Investigate the domain
  - √ Determine important concepts
  - √ Create formal representation
  - √ Populate knowledge base
- Carried out by *knowledge engineers* doing *knowledge acquisition*, e.g. interviewing domain experts
- The *declarative* (knowledge-based) approach has advantages over programming: can concentrate on stating *what* is true instead of worrying about *how* problems are solved (generic problem solvers)

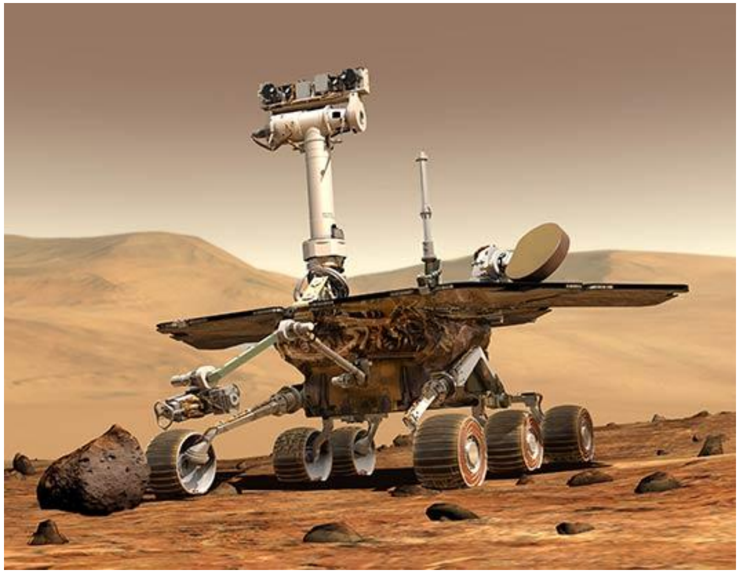
Ontologies

- An *ontology* is a vocabulary and a “theory” of a certain part of reality
- *Special-purpose* ontologies apply to restricted domains (e.g. electronics, drilling equipment, ..)
- *General-purpose* ontologies have wider applicability across domains



Planning as problem solving

Planning is a type of **problem solving**
in which the agent uses
beliefs about **actions** and their **consequences**
to find a **solution plan**,
where a plan is
a **sequence of actions**
that leads
from an **initial state**
to a **goal state**



Representing planning problems – PDDL

- ***Init****(At(C1, SFO) $\wedge$ At(C2, JFK) $\wedge$ At(P1, SFO) $\wedge$ At(P2, JFK) $\wedge$ Cargo(C1) $\wedge$ Cargo(C2) $\wedge$ Plane(P1) $\wedge$ Plane(P2) $\wedge$ Airport(JFK) $\wedge$ Airport(SFO))*
- ***Goal****(At(C1, JFK) $\wedge$ At(C2, SFO))*
- ***Action****(Load(c, p, a),*
  PRECOND: *At(c, a) $\wedge$ At(p, a) $\wedge$ Cargo(c) $\wedge$ Plane(p) $\wedge$ Airport(a),*
  EFFECT: $\neg$ *At(c, a) $\wedge$ In(c, p))*
- ***Action****(Unload(c, p, a),*
  PRECOND: *In(c, p) $\wedge$ At(p, a) $\wedge$ Cargo(c) $\wedge$ Plane(p) $\wedge$ Airport(a),*
  EFFECT: *At(c, a) $\wedge$ $\neg$ In(c, p))*
- ***Action****(Fly(p, from, to),*
  PRECOND: *At(p, from) $\wedge$ Plane(p) $\wedge$ Airport(from) $\wedge$ Airport(to),*
  EFFECT: $\neg$ *At(p, from) $\wedge$ At(p, to))*

Current planning approaches

- Forward state-space search with strong heuristics
- Planning graphs and GRAPHPLAN algorithm
- Partial order planning in plan space
- Planning as Boolean satisfiability (SAT)
- Planning as first-order deduction
- Planning as constraint-satisfaction

Planning in the real world

- Nondeterministic worlds
  - *Bounded* nondeterminism: Effects can be enumerated, but agent cannot know in advance which one will occur
  - *Unbounded* nondeterminism: The set of possible effects is unbounded or too large to enumerate
- Planning for bounded nondeterminism
  - Sensorless planning
  - Contingent planning
- Planning for unbounded nondeterminism
  - Online replanning
  - Continuous planning

Probability, utility and decisions

- The agent can use *probability theory* to reason about uncertainty
- The agent can use *utility theory* for rational selection of actions based on preferences
- *Decision theory* is a general theory for combining probability with rational decisions

Decision theory = Probability theory
+ Utility theory

Other approaches to uncertainty

- Default reasoning
- Rule-based methods
- Dempster-Shafer theory
- Fuzzy sets and fuzzy logic

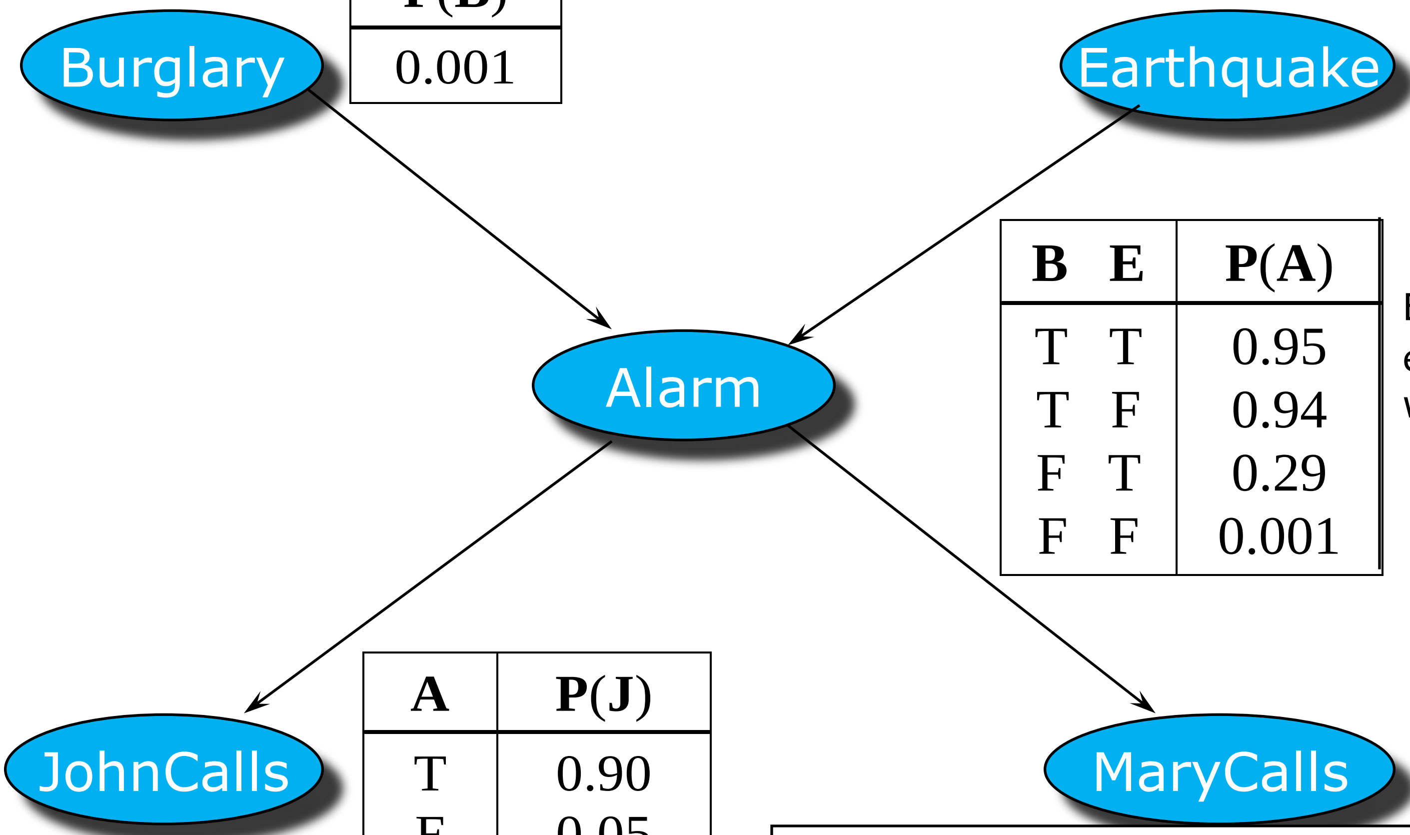
Bayes' rule

- Bayes' rule: $$P(B|A) = \frac{P(A|B)P(B)}{P(A)}$$
- Underlies modern AI systems for probabilistic inference
- Main application: How to use prior and causal knowledge (cause $\Rightarrow$ effect) to derive diagnosis (effect $\Rightarrow$ cause)

$$P(cause \mid effect) = \frac{P(effect \mid cause)P(cause)}{P(effect)}$$

- Use of causal knowledge is crucial in making probabilistic reasoning sufficiently robust in applications

Example Bayesian network



P(B)
0.001

P(E)
0.001

B	**E**	**P(A)**
T	T	0.95
T	F	0.94
F	T	0.29
F	F	0.001

Each CPT also has e.g. ¬ P(A) row with prob. 1-P(A)

A	**P(J)**
T	0.90
F	0.05

A	**P(M)**
T	0.70
F	0.01

Q: Given who calls, what is the prob. of Burglary?

Other uses of Bayesian networks

- *Make decisions* based on probabilities in the network and agent utilities (MEU)
- Decide which *additional evidence* variables should be observed in order to gain useful information
- Perform *sensitivity analysis* to understand which aspects of model have greatest impact on probabilities of query variables
- *Explain results* of probabilistic inference to users

Maximum Expected Utility (MEU)

- Let
  - *U*(*s*) - Utility of state *s*
  - *RESULT*(*a*) – Random variable whose values are possible outcome states of action *a* in current state
  - *P*(*RESULT*(*a*) = s′ | *a*, *e*) - Probability of outcome *s′*, as a result of doing action *a* in current state, and given agent's available evidence *e* of the world
- Then the *expected utility EU* of *a*, given *e* is

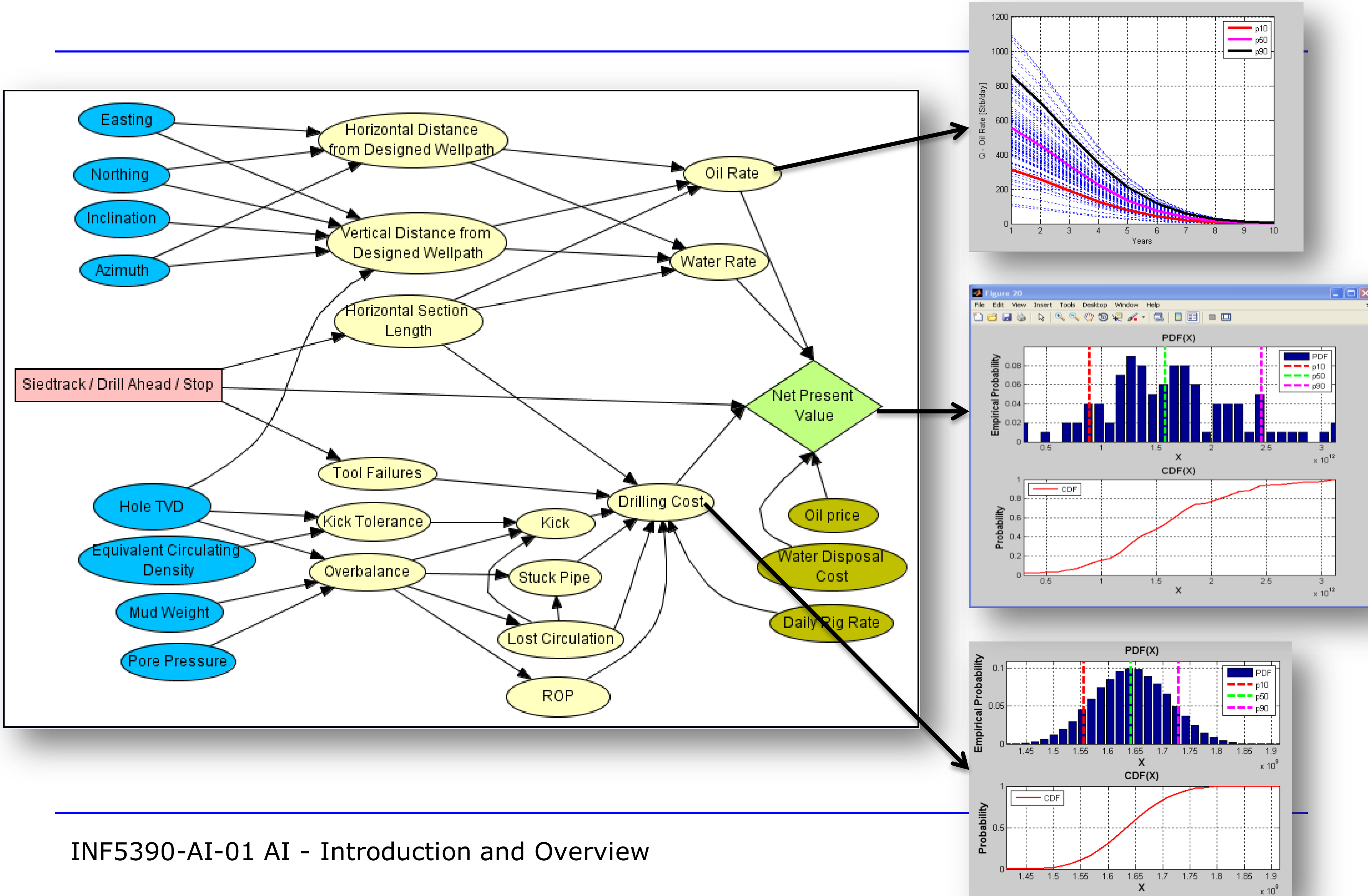
$$EU(a \mid e) = \sum_{s'} P(RESULT(a) = s' \mid a, e) U(s')$$

- MEU: Agent should select *a* that maximizes *EU*

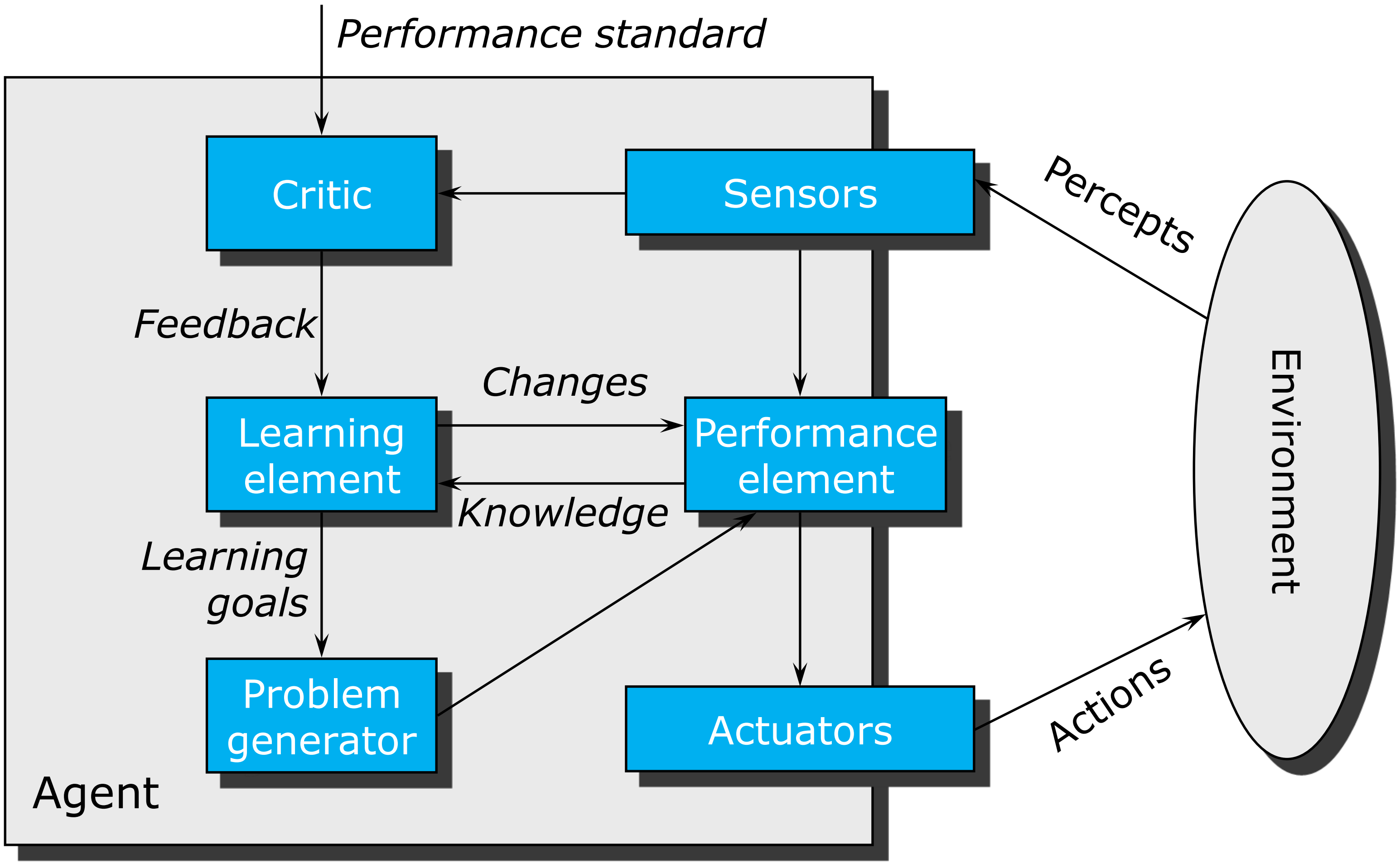
Decision networks

- *Decision networks* (also called *influence diagrams*) are a general mechanism for making rational decisions
- Decision networks combine belief networks with nodes for actions and utilities, and can represent
  - Information about agent's *current state*
  - Agent's *possible actions*
  - States that will *follow* from actions
  - *Utilities* of these states
- Therefore, decision networks provide a substrate for implementing rational, utility-based agents

Example decision network (CODIO)



Learning agents



What can be learned?

- Possible components that can be improved
  - √ Direct mapping from states to actions
  - √ Infer world properties from percept sequences
  - √ Information about how the world evolves
  - √ Information about the results of possible actions
  - √ Utility information about the desirability of world states
  - √ Desirability of specific actions in specific states
  - √ Goals describing states that maximize utility
- In each case, learning can be sees as learning an unknown *function* $y = f(x)$

Types of learning - Knowledge

- *Inductive* learning
  - Given a collection of *examples* $(x, f(x))$
  - Return a function *h* that approximates *f*
  - Does not rely on prior knowledge (“just data”)
- *Deductive* (or analytical) learning
  - Going from known general *f* to a new *f′* that is logically entailed
  - Based on prior knowledge (“data+knowledge”)
  - Resembles more human learning

Types of learning - Feedback

- *Unsupervised* learning
  - √ Agent learns patterns in data even though no feedback is given, e.g. via clustering
- *Reinforcement* learning
  - √ Agent gets reward or punishment at the end, but is not told which particular action led to the result
- *Supervised* learning
  - √ Agent receives learning examples and is explicitly told what the correct answer is for each case
- Mixed modes, e.g. *semi-supervised* learning
  - √ Correct answers for some but not all examples

Learning decision trees

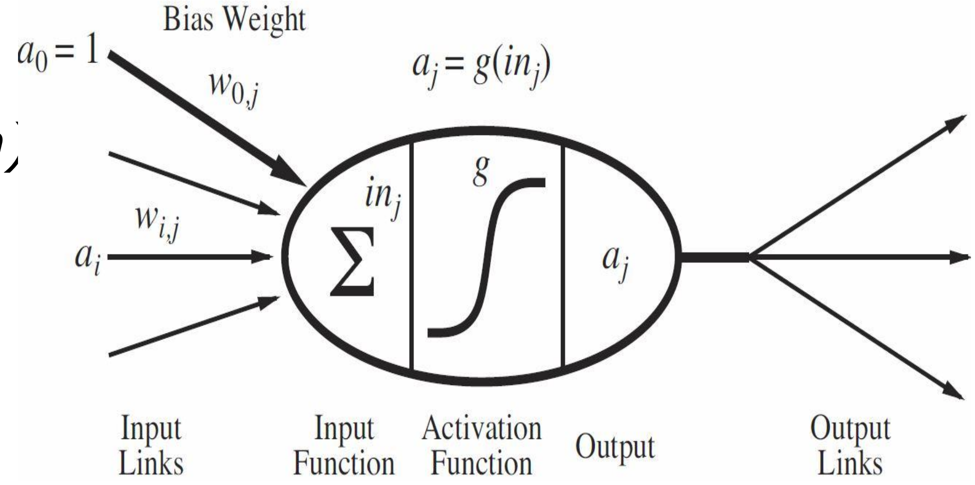
- A *decision situation* can be described by
  - A number of *attributes*, each with a set of possible values
  - A *decision* which may be Boolean (yes/no) or multivalued
- A *decision tree* is a tree structure where
  - Each internal node represents a *test* of the value of an attribute, with one branch for each possible attribute value
  - Each leaf node represents the value of the *decision* if that node is reached
- *Decision tree learning* is one of simplest and most successful forms of machine learning
- An example of *inductive* and *supervised* learning

Other knowledge-based learning methods

- EBL – Explanation-based learning
  - √ Extracts general rules from single examples accompanied by an explanation
- RBL – Relevance-based learning
  - √ Uses prior knowledge to identify relevant attributes thereby reducing hypothesis space
- KBIL – Knowledge-based inductive learning
  - √ Uses prior knowledge to find inductive hypotheses that explain sets of observations
- ILP – Inductive logic programming
  - √ Performs KBIL on knowledge expressed in first-order logic, and can learn relational knowledge

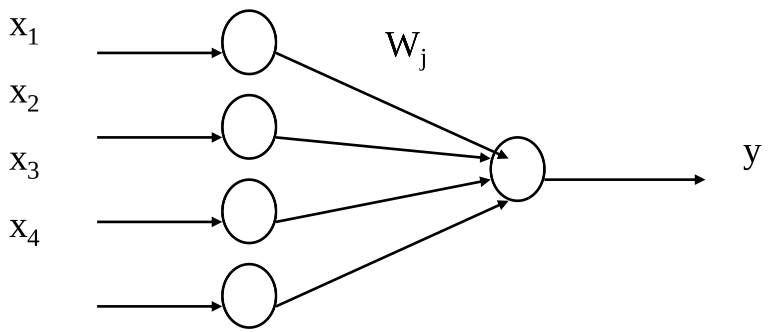
Neural networks

- The network consists of *units* (nodes, "neurons") connected by *links*
  - Carries an *activation* a_i from unit i to unit j
  - The link from unit i to unit j has a *weight* $W_{i,j}$
  - *Bias* weight $W_{0,j}$ to fixed input $a_0 = 1$
- *Activation* of a unit j
  - Calculate input $in_j = \sum W_{i,j}\, a_i \quad (i=0..n)$
  - Derive output $a_j = g(in_j)$ where g is the *activation function*



Learning in neural networks

- How to train the network to do a certain function (e.g. *classification*) based on *a training set* of input/output pairs?



- Basic idea
  - Adjust network link weights to minimize some measure of the error on the training set
  - Adjust weights in direction that minimizes error

Genetic algorithms (GA)

- *Genetic algorithms* is a search method that "evolves" a solution by a process of natural selection in a population of solutions, inspired by natural evolution

```
function GENETIC-ALGORITHM(population, FITNESS-FN)
    returns an individual
    inputs: population, a set of individuals
                FITNESS-FN, a function that measures individual fitness
      repeat
          parents <= SELECTION(population, FITNESS-FN)
          population <= REPRODUCTION(parents)
      until some individual is fit enough
      return the best individual in population, according to FITNESS-FN
```

Properties of GA

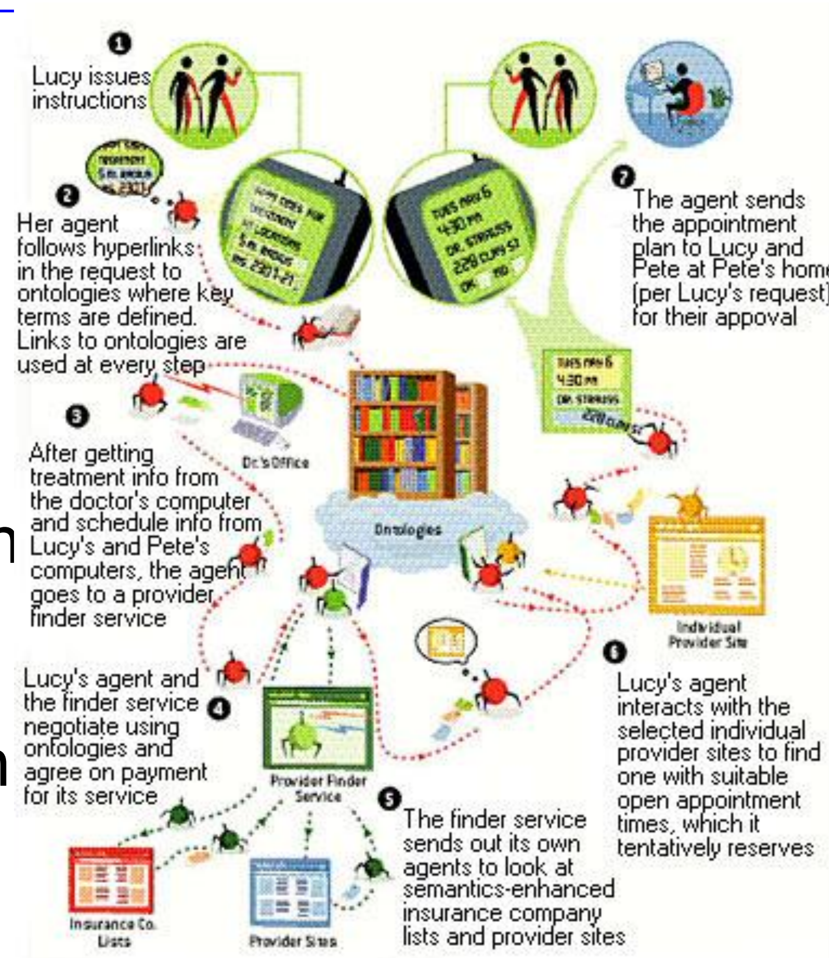
- GA are easy to program and apply, but optimal results are not guaranteed
- GA search is *parallel*, since each individual is a search by itself
- The search is *hill climbing*, since small genetic changes are used to explore space
- The search *avoids local maxima* by using “mutation” jumps
- It can be shown that GA allocates search resources in an optimal way (under certain assumptions)

Semantic Web – AI for the Internet?

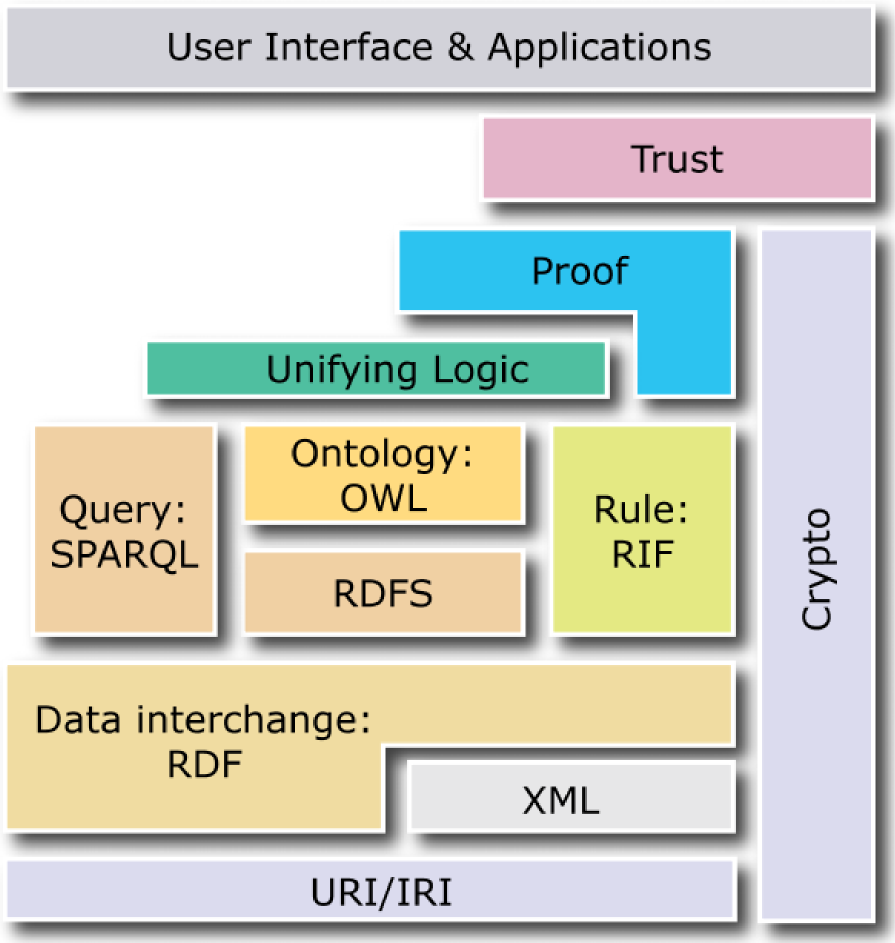
- The classical Web lacks semantics and requires human users to interpret content
- The Semantic Web adds metadata to Web contents and is machine processable
- The technological basis for the Semantic web are metadata, ontologies, logic, and agents
- Implementation of the Semantic web follows a layered set of representation languages

Semantic Web – Original vision

- Agents collect information from Web-pages
- Use semantics to disambiguate data
- Process information using logical inference
- Exchange information with and negotiate with other agents
- Automate routine problem solving



Semantic Web «layer cake»



Recapitulation: AI as agent design

- The AI “project” can be seen as the design of *intelligent agents*
- Different agent designs are possible, from *reflex* agents to *deliberative* knowledge-based ones
- Different paradigms are being used: logical, probabilistic, “neural”
- Do we have the necessary tools to build *a complete, general-purpose agent*?

Uneven status of AI disciplines

- Some parts of AI are *mature*, and agents can be built that outperform humans in these areas
  - E.g.: Game playing, logical inference, theorem proving, planning, diagnosis
- Other parts of AI are *evolving*, where progress is being made
  - E.g.: Learning, vision, robotics, natural language understanding

AI - Summary

- AI is a cross disciplinary profession and can be defined as the theory of how to build intelligent agents
- Intelligent agents must be able to solve problems, possess knowledge, be able to reason, plan, handle uncertainty, learn, and communicate
- AI is being applied to solve problems of practical and economic value
- AI has a 50+ year history, is in continuous evolution, and has an open future!