Knowledge Representation

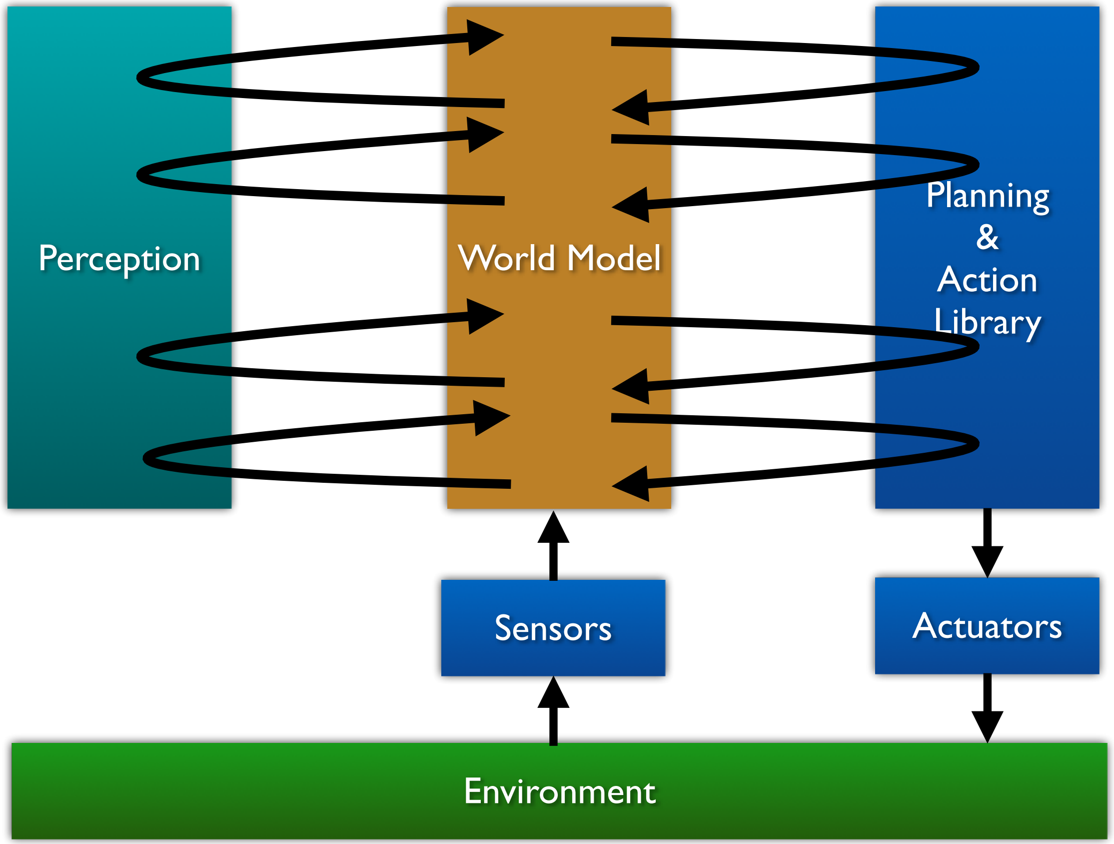
COMP3411/9814: Artificial Intelligence

Lecture Overview

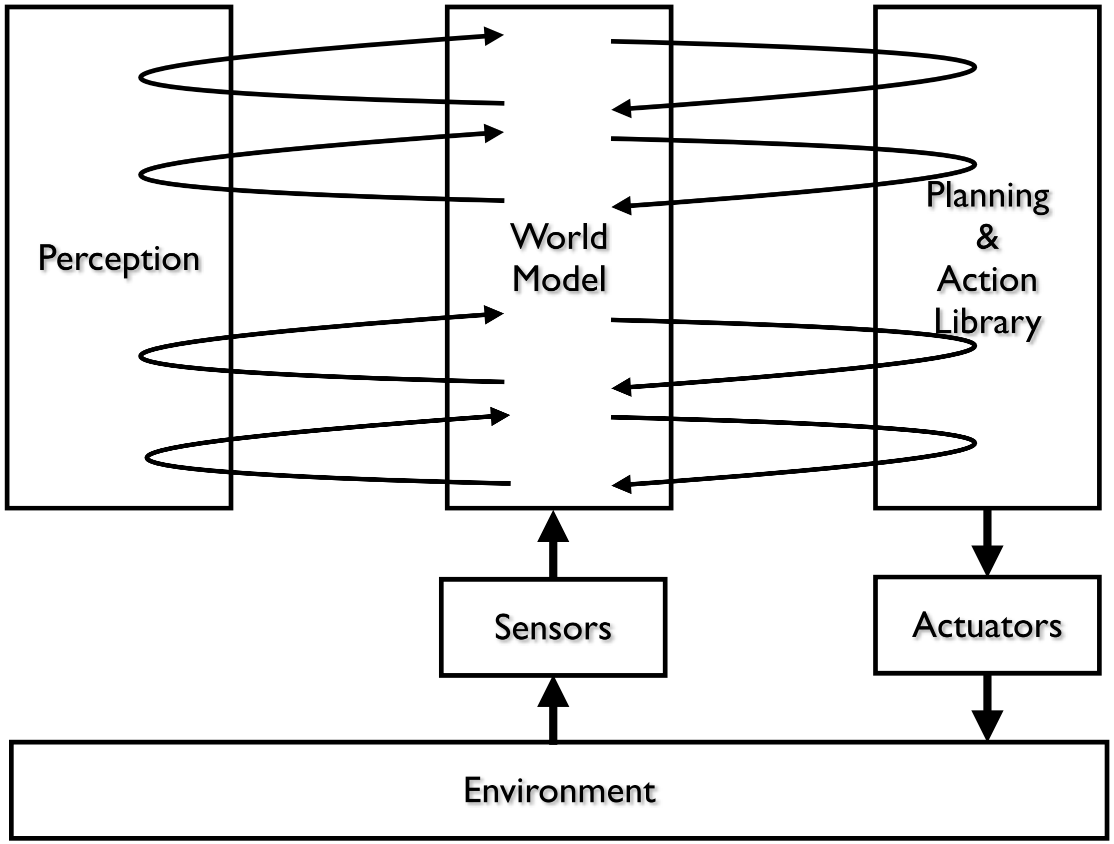
- Cognitive Architectures
- The Knowledge Level
- Knowledge Representation
- Ontologies, Taxonomy, Categories and Objects
- Semantic Networks
- Rule based representation
- Inference Networks
- Deduction, Abduction, and Induction

Cognitive Architectures

Nilsson’s Triple Tower

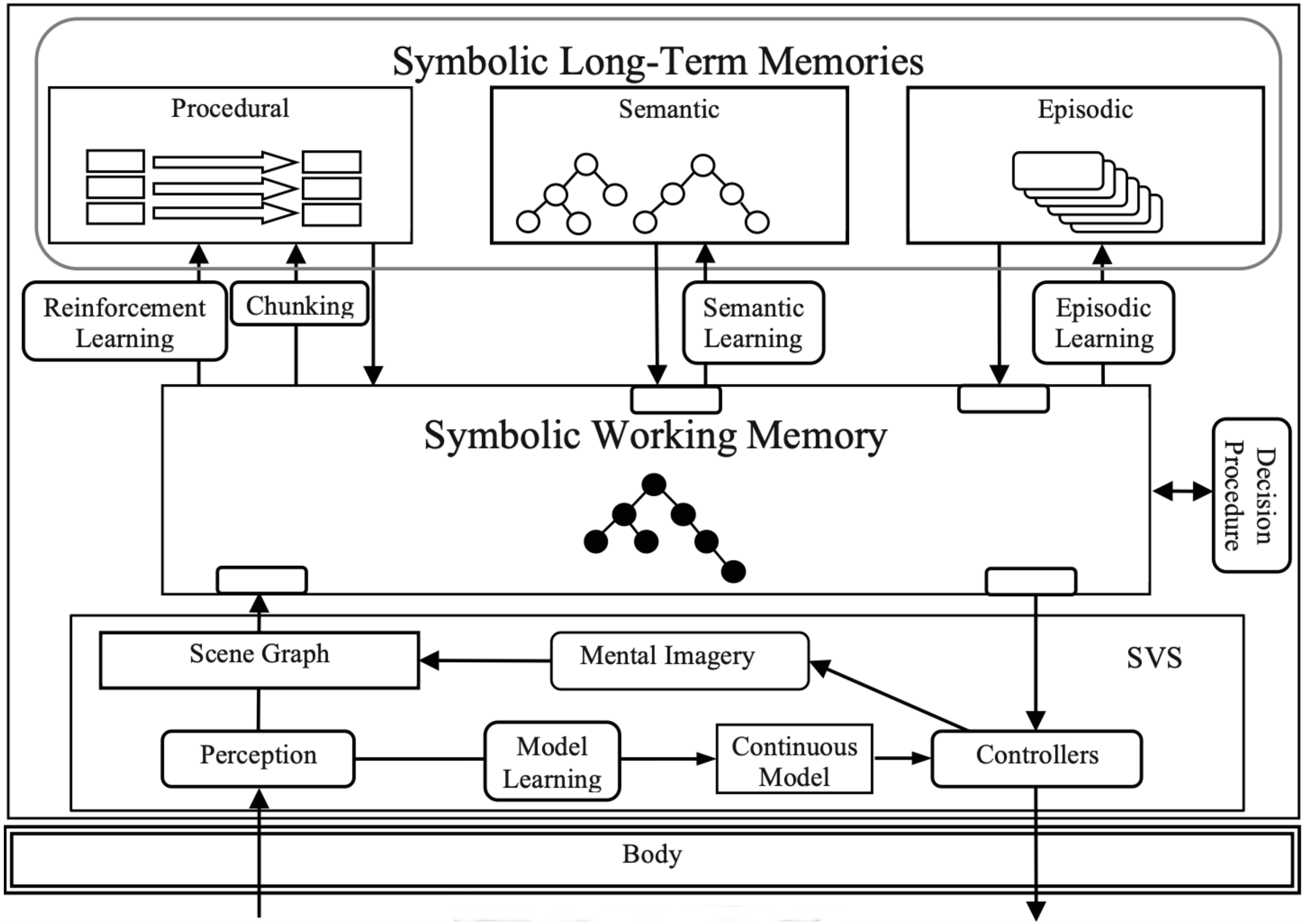


Nilsson's Triple Tower

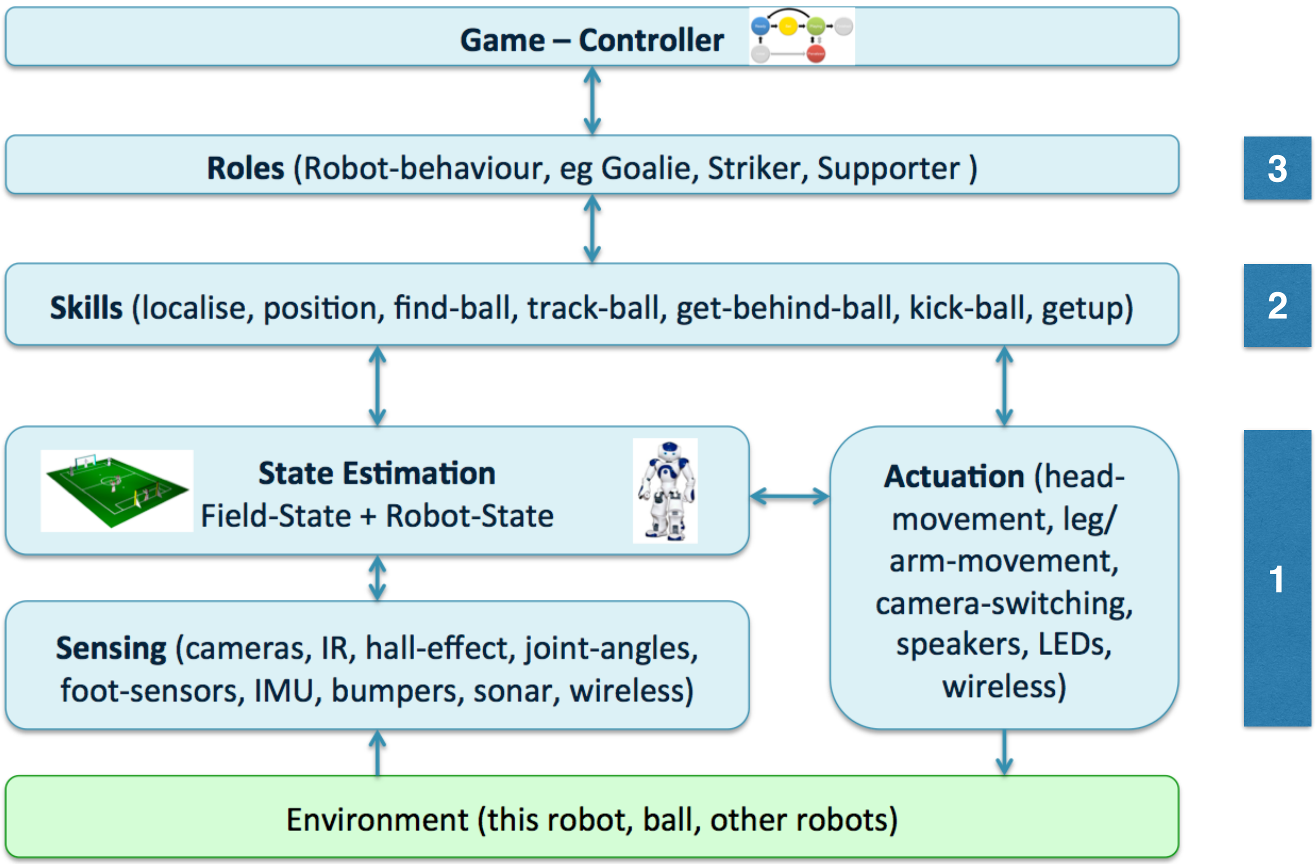


SOAR

Laird, J. E., Kinkade, K. R., Mohan, S., & Xu, J. Z. (2012). Cognitive Robotics Using the Soar Cognitive Architecture (pp. 46–54). In *International Conference on Cognitive Robotics, (Cognitive Robotics Workshop, Twenty-Sixth Conference on Artificial Intelligence (AAAI-12)*, Toronto.



A Three-Level Architecture



“Classical” AI

Symbolic Representations and Reasoning

The Physical Symbol System Hypothesis

"A physical symbol system has the necessary and sufficient means for general intelligent action."[1]

— Allen Newell and Herbert A. Simon

Criticisms:

- Lacks “symbol grounding” (what does a symbol refer to?).
- AI requires non-symbolic processing (e.g. connectionist architecture).
- Brain is not a computer and computation is not an appropriate model for intelligence.
- Brain is mindless - mostly chemical reactions
  - human intelligent behaviour is analogous to behaviour displayed by ant colonies

The Knowledge Level

- Knowledge Level Hypothesis. There exists a distinct computer systems level, lying immediately above the symbol level, which is characterised by knowledge as the medium and the principle of rationality as the law of behaviour.

- Principle of Rationality. If an agent has knowledge that one of its actions will lead to one of its goals, then the agent will select that action.

- Knowledge. Whatever can be ascribed to an agent, such that its behaviour can be computed according to the principle of rationality.

“The Knowledge Level” (Newell, 1982)

Knowledge Representation

- Any agent can be described on different levels:
  - Knowledge level (knowledge ascribed to agent)
  - Logical level (algorithms for manipulating knowledge)
  - Implementation level (how algorithms are implemented)
- Knowledge Representation is concerned with expressing knowledge explicitly in a computer-tractable way (i.e. for reasoning)
- Reasoning takes knowledge and draw inferences
  - answer queries, determine facts that follow from the knowledge, decide what to do, etc.

Knowledge Representation and Reasoning

- A knowledge-based agent has at its core a knowledge base
- A knowledge base is an explicit set of sentences about some domain expressed in a suitable formal representation language
- Sentences express facts (true) or non-facts (false)
- Fundamental Questions
  - How do we write down knowledge about a domain/problem?
  - How do we automate reasoning to deduce new facts or ensure consistency of a knowledge base?

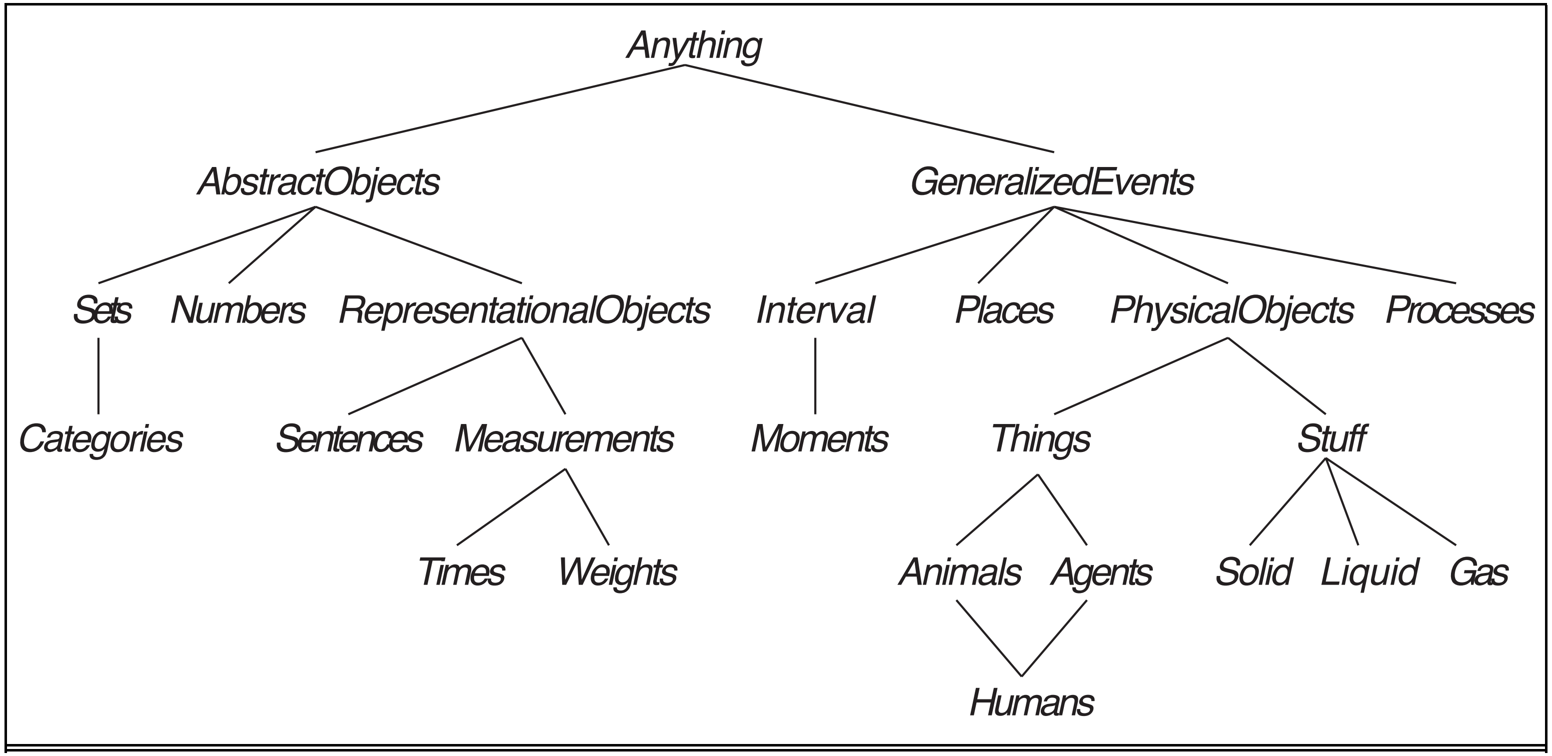
Knowledge representation

- We study the technology for knowledge-based agents:
  - syntax, semantics and proof theory of propositional and first-order logic
  - the implementation of agents that use these logics.
- Also need to address question:
  - What *content* to put into such an agent's knowledge base?
  - how to represent facts about the world.

Ontologies and Ontology Engineering

- An ontology organises everything into a hierarchy of categories.
- Can't actually write a complete description of everything
  - far too much
  - can leave placeholders where new knowledge can fit in.
  - define what it means to be a physical object
  - details of different types of objects (robots, televisions, books, …) filled in later
- Similar to Object Oriented programming framework

Ontology Example



- Child concept is a specialisation of parent
- Specialisations are not necessarily disjoint (a human is both an animal and an agent)

Ontology Example

- Equality: Scott Morrison is Prime Minister Morrison
- Role: Scott Morrison is Prime Minister of Australia
- Part of: Scott Morrison is in the government
- A kind of: NSW is a state
- Part of: NSW is in Australia

- Vaccination implies Medial treatment – linguistic meaning/semantics

- Ontology = Set of such facts

Categories and Objects

- Organising objects into categories is vital for knowledge representation.
- Interaction with world takes place at level of individual objects, but …
  - much reasoning takes place at level of categories
- Categories help make predictions about objects once they are classified
- Two choices for representing categories in first-order logic:
  - predicates and objects.

Categories and Objects

- Infer presence objects from perceptual input
- Infer category membership from perceived properties of the objects
- Use category information to make predictions about the objects
  - green and yellow mottled skin & 30cm diameter & ovoid shape & red flesh, black seeds & presence in the fruit aisle → watermelon
  - from this, infer that it would be useful for fruit salad.

Categories and Objects

- Categories organise and simplify knowledge base through inheritance.
  - if all instances of category Food are edible, and
  - if Fruit is a subclass of Food and Apples is a subclass of Fruit, then
  - infer that every apple is edible.
- Individual apples inherit property of edibility
  - in this case from membership in the Food category.

Taxonomic hierarchy

- Subclass relations organise categories into a taxonomy, or taxonomic hierarchy
- Taxonomies have been used explicitly for centuries in technical fields.
  - Taxonomy of living things organises about 10 million living and extinct species
  - Library science has developed a taxonomy of all fields of knowledge
- Taxonomies are also an important aspect of general commonsense knowledge

Categories and Objects and FOL

- First-order logic can state facts about categories, relating objects to categories or by quantifying over members.
- An object is a member of a category

$$BB_9 \in Basketballs$$

- A category is a subclass of another category

$$Basketballs \subset Balls$$

- All members if a category have some properties

$$(\forall x)(x \in Basketballs \implies Spherical(x))$$

- Members of a category can be recognised by some properties

$$Orange(x) \land Round(x) \land Diameter(x) = 24cm \land x \in Balls \implies x \in Basketballs$$

- A category as a whole has some properties

$$Dogs \in DomesticatedSpecies$$

```
Prolog:

  basketball(X) :-
    orange(X),
    round(X),
    diameter(X, 24),
    ball(X).
```

Reasoning system for categories

- Categories are the building blocks of knowledge representation schemes
- Two closely related families of systems:
  - semantic networks:
    - graphical aids for visualizing a knowledge base
    - efficient algorithms for inferring properties of object based in category membership
  - description logics:
    - formal language for constructing and combining category definitions
    - efficient algorithms for deciding subset and superset relationships between categories

Semantic Networks

- Fact , Objects, Attributes and Relationships
  - Relationships exist among instances of objects and classes of objects.
- Attributes and relationships can be represented as a network, known as an associative network or semantic network
- We can build a model of the subject area of interest

Semantic networks

- In 1909, Charles S. Peirce proposed a graphical notation of nodes and edges called existential graphs that he called “the logic of the future.”
- Long-running debate between advocates of “logic” and “semantic networks”:
  - semantics networks with well-defined semantics are a form of logic.
  - notation provided by semantic networks for certain kinds of sentences is often more convenient,
  - underlying concepts
    - objects, relations, quantification, and so on…
    - are the same as in logic.

Knowledge and Semantic Networks

- Facts can be:
  - ***static*** - can be written into the knowledge base.

    static facts need not be permanent, but change infrequently so changes can be accommodated by updating the knowledge base when necessary.

- ***transient*** - apply at a specific instance only or for a single run of system

Knowledge and Semantic Networks

- Important aspect of semantic networks - can represent default values for categories

- Knowledge base may contain *defaults* that can be used as facts in the absence of transient facts

Example – A simple set of statements

- My car is a car
- A car is a vehicle
- A car has four wheels
- A car’s speed is 0 mph
- My car is red
- My car is in my garage
- My garage is a garage
- A garage is a building
- My garage is made from brick
- My car is in High Street
- High Street is a street
- A street is a road

Underline = object (instance)
Everything else is a category (class)

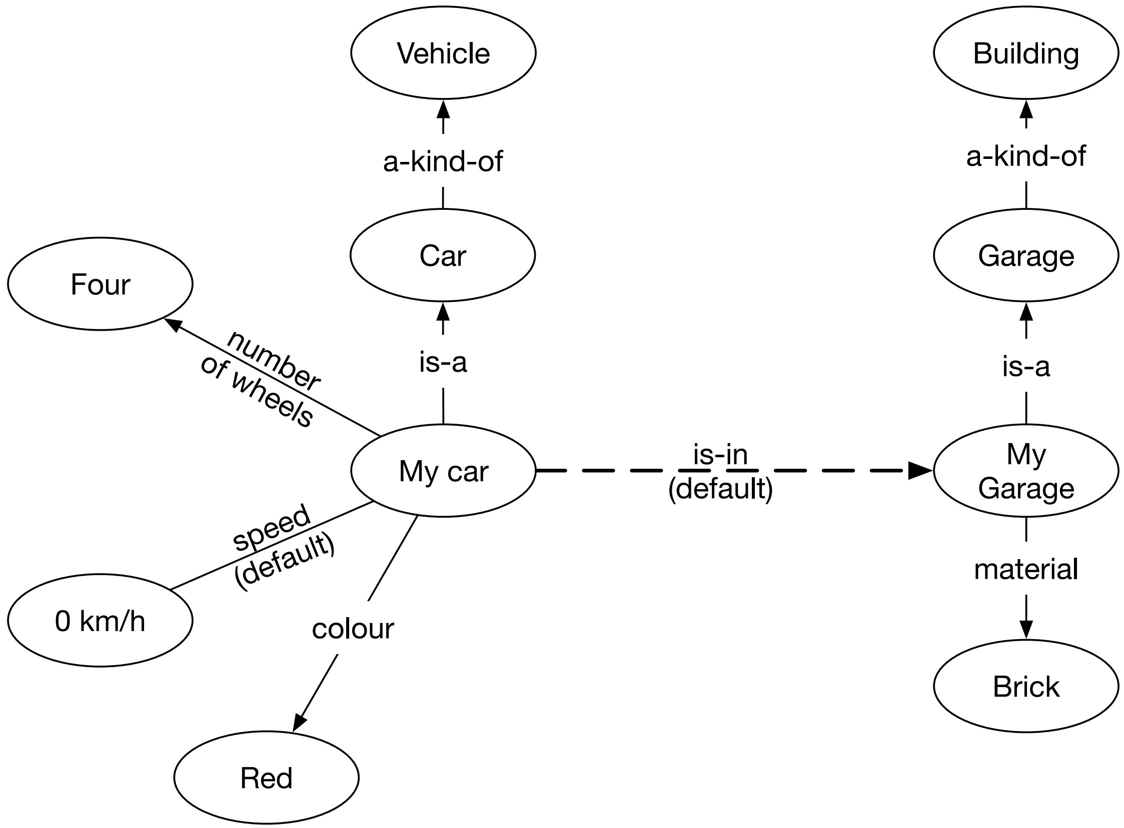
Example – facts, objects and relations

- My car is a **car**
- A car is a **vehicle**
- A car has four wheels
- A car’s speed is 0 mph
- My car is red
- My car is in my garage
- My garage is a **garage**
- A garage is a **building**
- My garage is made from brick
- My car is in High Street
- High Street is a **street**
- A street is a **road**

Example – facts, objects and relations

- My car is a **car** (static relationship)
- A car is a **vehicle** (static relationship)
- A car has four wheels (static attribute)
- A car's speed is 0 mph (default attribute)
- My car is red (static attribute)
- My car is in my garage (default relationship)
- My garage is a **garage** (static relationship)
- A garage is a **building** (static relationship)
- My garage is made from brick (static attribute)
- My car is in High Street (transient relationship)
- High Street is a **street** (static relationship)
- A street is a **road** (static relationship)

A semantic network (with a default)



Classes and Instances

- Distinction between object instances and classes of objects:
  - Car and vehicle are both classes of objects
    - Linked by “ako” relation (a-kind-of)
    - Direction of arrow indicates “car is a vehicle” and not “vehicle is a car”
- *My car* is a unique entity.
  - Relationship between *my car* and *car* is “isa” (is an instance of)

Semantic Networks - Reasoning

- Inheritance is good for default reasoning weak otherwise
- Extend by *procedural attachment*
  - **Frames:** Demons are triggered when attributes if instances are added, deleted or modified
  - **Agents:** Contain gaols and plans and run as concurrently
  - **Objects:** Methods an implement attached procedures

Rule-Based Systems

- A production rule and has the form

```
if <condition> then <conclusion>
```

- Production rule for dealing with the payroll of ACME, Inc., might be

```
rule r1_1
if the employer of Person is acme
then the salary of Person becomes large.
```

Rule-Based Systems

```
rule r1_1

if the employer of Person is acme

then the salary of Person becomes large.
```

- Production rules can often be written to closely resemble natural language

```
/* fact f1_1 */

the employer of joe_bloggs is acme.
```

- Capitalisation (like Prolog) indicates that “Person” is a variable that can be replaced by a constant, such as “joe_bloggs” or “mary_smith”, through pattern matching.

Rule-Based Systems

```
rule r1_1
if the employer of Person is acme
then the salary of Person becomes large.
```

```
rule r1_2
if the salary of Person is large
or the job_satisfaction of Person is true
then the professional_contentment of Person becomes true.
```

- Executing a rule may generate a new derived fact.
- There is a *dependency* between rules r1_1 and r1_2 since the conclusion of one can satisfies the condition of the other.

Uncertainty in rules

- Rules can express many types of knowledge
- But how can *uncertainty* be handled?
- Uncertainty may arise from:
  - Uncertain evidence (Not certain that Joe Bloggs works for ACME.)
  - Uncertain link between evidence and conclusion.
    (Cannot be certain that ACME employee earns a large salary, just likely.)
  - Vague rule. (What is a “large”?)

Fuzzy Logic

Bayesian inference

Rule-Based Systems

```
/* fact f1_1 */

the employer of joe_bloggs is acme.
```

```
rule r1_1

if the employer of Person is acme

then the salary of Person becomes large.
```

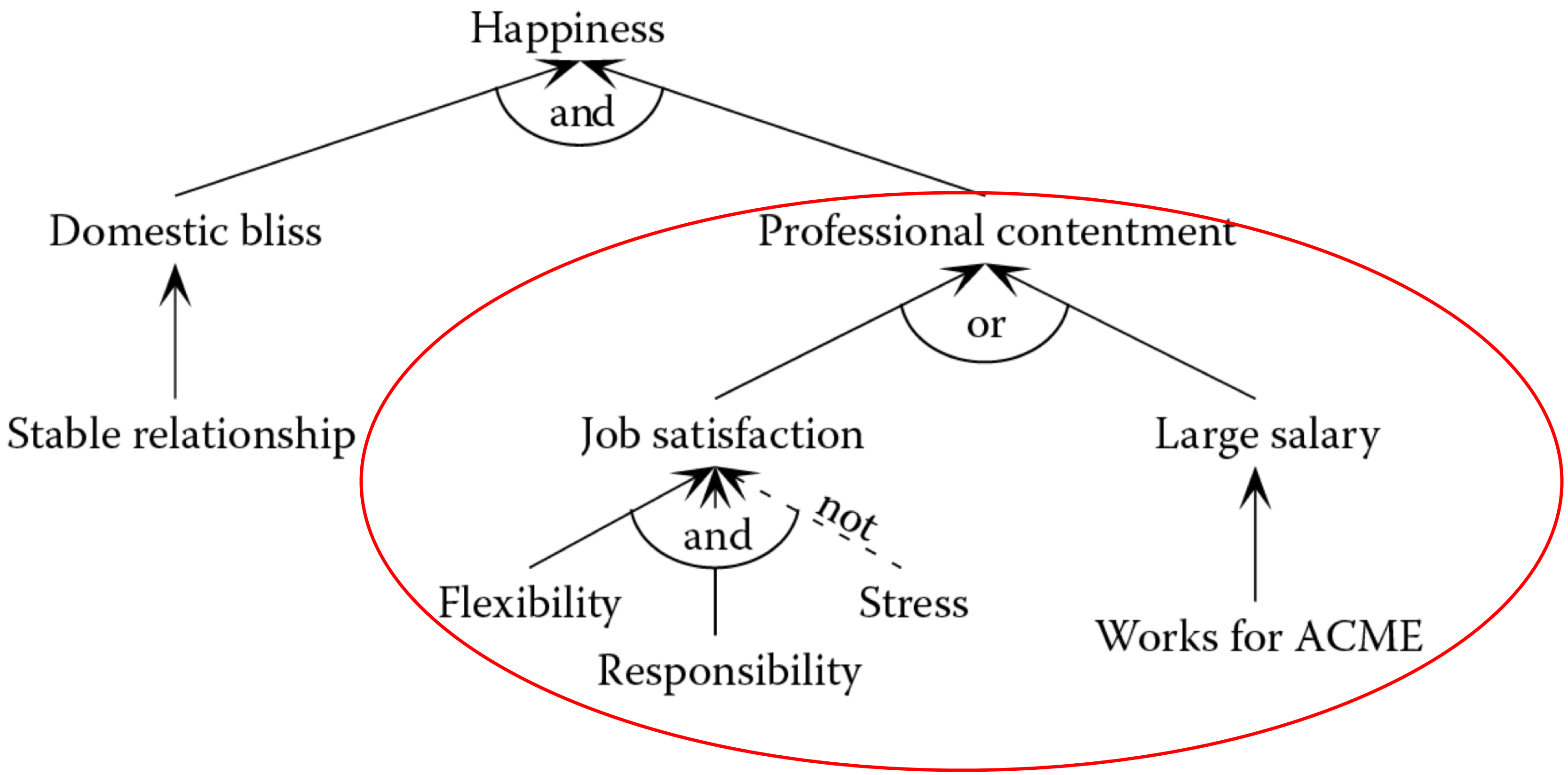
```
/* derived fact f1_2 */

the salary of joe_bloggs is large.
```

```
rule r1_2

if the salary of Person is large

or the job_satisfaction of Person is true

then the professional_contentment of Person becomes true.
```

Inference Networks

- The interdependencies among rules, such as r1_1 and r1_2 define a network
- Inference network shows which facts can be logically combined to form new facts or conclusions
- The facts can be combined using “and”, “or” and “not”.
    - Professional contentment is true if either job satisfaction or large salary is true (or both are true).
    - Job satisfaction is achieved through flexibility, responsibility, and the absence of stress.

An Inference Network



Professional contentment is true if either job satisfaction or large salary is true (or both are true).

An Inference Network

- An inference network can be constructed by
  - taking *facts* and working out what conditions have to be met for those facts to be true.
  - After these conditions are found, they can be added to the diagram and linked by a *logical expression* (such as and, or, not).
  - This usually involves breaking down a complex logical expression into smaller parts.

Deduction, Abduction and Induction

Rules that make up inference network can be used to link cause and effect:

if <cause> then <effect>

E.g.:

if

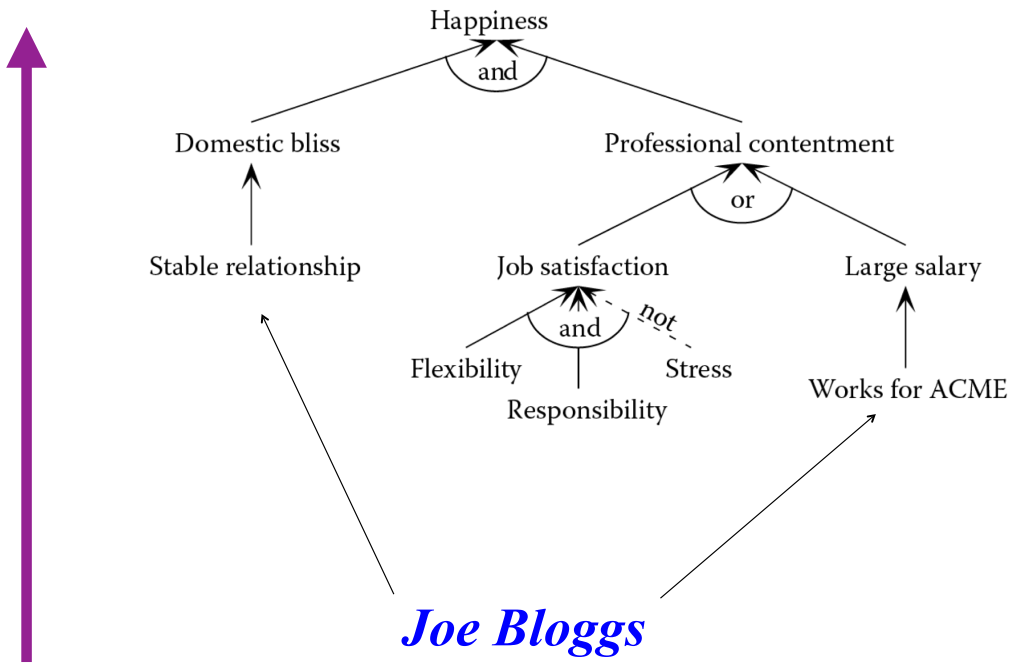
Joe Bloggs works for ACME and is in a stable relationship (the causes),

then

he is happy (the effect).

Deduction, Abduction and Induction

if <cause> then <effect>

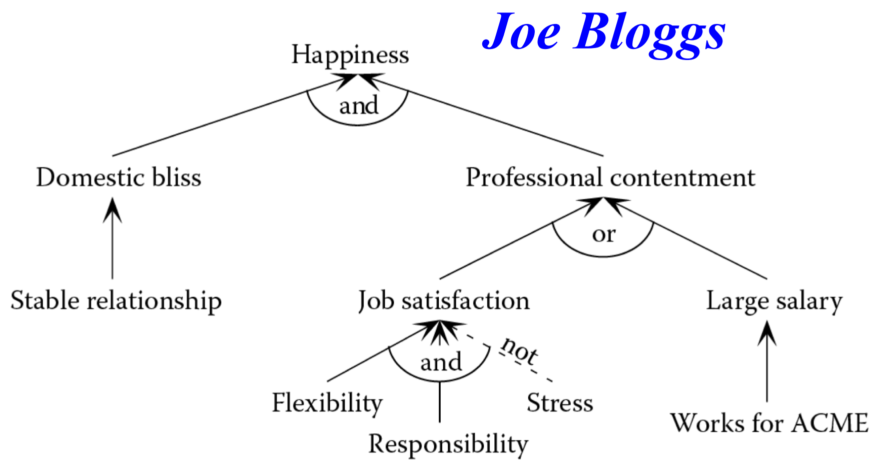


if *Joe Bloggs* works for ACME and is in a stable relationship (causes), then he is happy (effect).

Deduction, Abduction and Induction

- Abduction - Many problems, such as diagnosis, involve reasoning in reverse, i.e, find a cause, given an effect.

- Given observation Joe Bloggs is happy, infer by abduction Joe Bloggs enjoys domestic bliss and professional contentment.



Deduction, Abduction and Induction

- If we have many examples of cause and effect, infer the rule that links them.
- E.g, if every employee of ACME earns a large salary, infer:

```
rule r1_1
  if the employer of Person is acme
  then the salary of Person becomes large.
```

- Inferring a rule from a set of examples of cause and effect is induction.

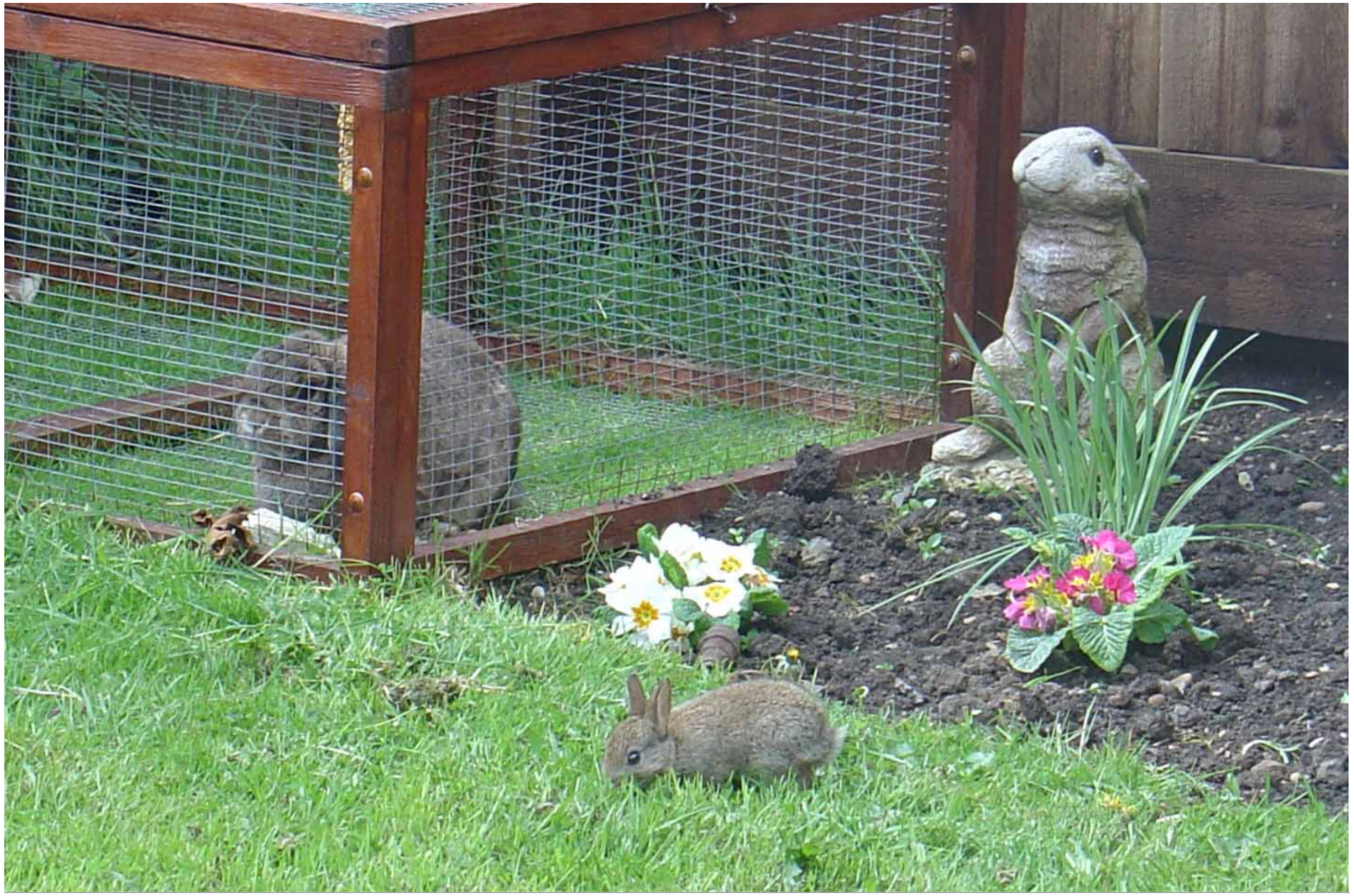
Deduction, Abduction and Induction

- deduction: cause + rule $\Rightarrow$ effect
- abduction: effect + rule $\Rightarrow$ cause
- induction: cause + effect $\Rightarrow$ rule

Closed-World Assumption

- Only facts that are in the knowledge base or that can be derived from rules are assumed to be true
- Everything is assumed to be false
- I.e. if we don't know it, it's assumed to be false
- That's why it's more accurate to say:
    - "a proof fails", instead of "it's false"
    - "a proof succeeds" instead of, "it's true"

How many rabbits are there?



How many rabbits are there?

- Perception isn't all in the eye.
- Knowledge is usually needed to understand the world

References

- Poole & Mackworth, Artificial Intelligence: Foundations of Computational Agents, Chapter 5

- Russell & Norvig, *Artificial Intelligence: a Modern Approach*, Chapter 12.

- Adrian A. Hopgood. *Intelligent Systems for Engineers and Scientists (3rd Edition)*. CRC Press, 2011 – Chapter 1.