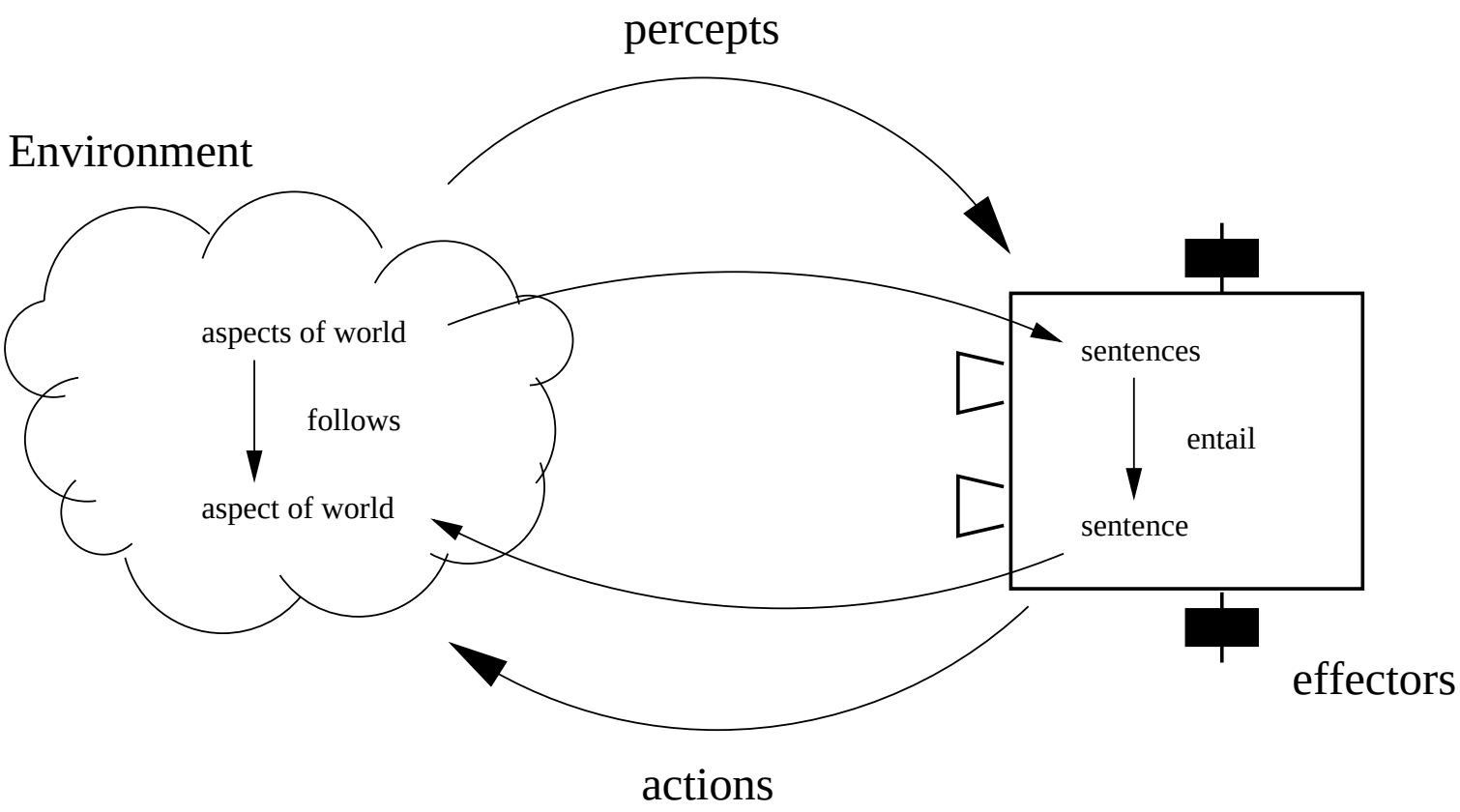
KNOWLEDGE REPRESENTATION

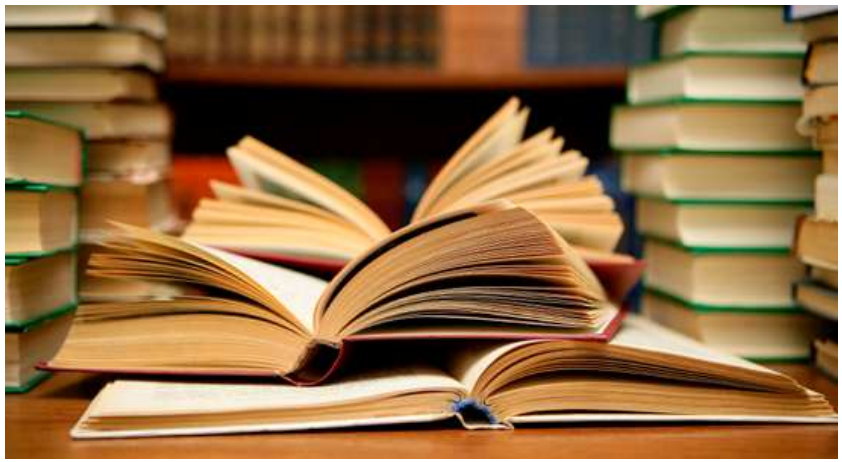
Introduction

- For the last couple of weeks we talked about logic.
- In particular we talked about propositional logic and first order logics.
- The claim was that we could have agents use them to reason about the world.
  - Implies *representation*
    (see over)
- This week we'll look at using logic for representation.
- As we'll see, it opens up a range of new problems.


percepts
Environment
aspects of world
follows
aspect of world
sentences
entail
sentence
effectors
actions

Knowledge Engineering

- Is there a general process for writing down a description of the world in logic.
  - Kind of
- The textbook gives the following steps, which make sense.



- Identify the task

  First you decide what kinds of query the system will have to answer. You don't want to miss things out, or waste time including things that are unecessary.

- Assemble the relevant knowledge.

  Obtain an informal understanding of the domain, if you don't have one.

- Decide on an *ontology*.

  Translate the domain concepts into logical predicates. How do you represent movement?

- Encode general knowledge about the domain.

  Now you write down the ontology in logic, writing axioms for all the terms in the vocabulary.

- Encode a specific problem instance

  If the ontology is correct, domain knowledge is largely simple statements from the ontology. If not, this is when you start to find that the ontology is wrong.

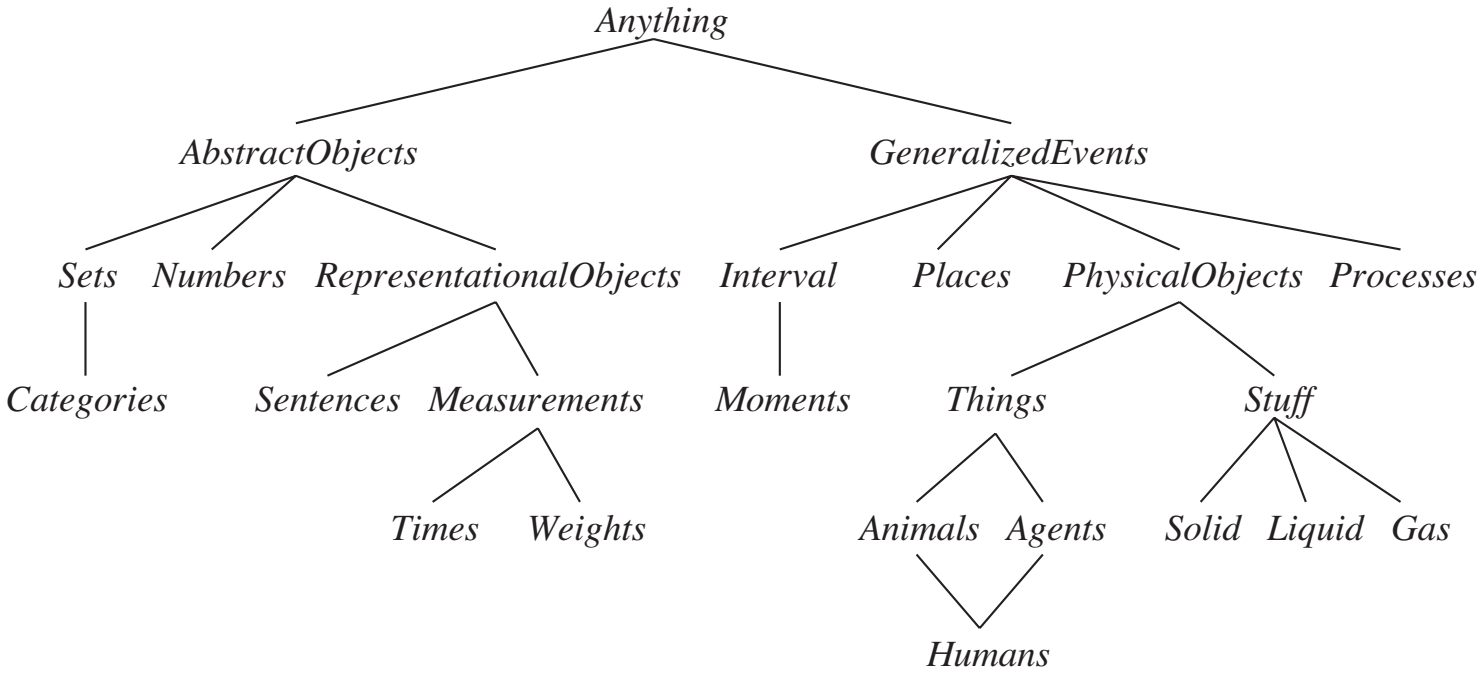
- Pose queries

  This is the testing step — the inference mechanism should generate correct answer to your queries, when it doesn't you go on to...

- Debug the knowledge base

  Usually you will find that the ontology has flaws in it somewhere. Axioms missing, or statements that are too weak (too few inferences sanctioned) or too strong (too many inferences sanctioned).

Ontologies



- Many aspects of the world are hard to formalise in logic.

Categories and objects

- To build an ontology we must divide objects into categories
  - We interact at the individual objects, but think in categories.
  - "I'm going to buy a baseball", not "I'm going to buy $baseball_{2367541.885}$
- Categories are also useful.
  - Define properties of categories and then have objects inherit them.

$$Green\_mottled(x) \Rightarrow Watermelon(x) \Rightarrow Good\_to\_eat(x)$$

  - Infer category membership and make predictions.

- Can represent categories as predicates

$$Basketball(b)$$

- Can also represent categories as objects, *reifying* them.

$$Basketballs$$

  so we can then say:

$$Member(b, Basketballs) \text{ or } b \in Basketballs$$

- Also have subset relationships:

$$Subset(Basketballs, Balls) \text{ or } Basketballs \subset Balls$$

- Here *Basketballs* is subset, *subclass* and/or *subcategory*.

Subclasses

- We organise categories by subclass relations, and use *inheritance* to save on representation.
- $b \in$ *Basketballs* will be round, because *Basketballs* $\subset$ *Balls* and balls are round.

  (Well, of course they aren't *all* round, but there are inheritable properties).

- Subclasses organise knowledge into a taxonomic hierarchy.

- Statements about categories are typically easy to make using first order logic:

$$bb_9 \in \mathit{Basketballs}$$
$$\mathit{Basketballs} \subset \mathit{Balls}$$
$$(x \in \mathit{Basketballs}) \Rightarrow \mathit{Spherical}(x)$$

- This last shows one way to restrict the domain of a universal quantification.
- Can also make statements about members of a category, and categories as a whole.

$$\mathit{Orange}(x) \wedge \mathit{Diameter}(x) = 9.5'' \wedge x \in \mathit{Balls} \Rightarrow x \in \mathit{Basketballs}$$
$$\mathit{Dogs} \in \mathit{DomesticatedSpecies}$$

- We also want to be able to state things that are not sub-class relationships.

Other properties

- Categories are *disjoint* they have no members in common.

$$Disjoint(\{Animals, Vegetables\}).$$

- An *exhaustive decomposition* lists all of the possible constituents:

$$ExhaustiveDecomposition(\{Americans, Canadians, Mexicans\}, NorthAmericans)$$

- If an exhaustive decomposition is disjoint, it is a *partition*

$$Partition(\{Males, Females\}, Animals)$$

Physical composition

- The *PartOf* relation is also important.

$$PartOf(Bucharest, Romania)$$
$$PartOf(Romania, Europe)$$
$$PartOf(Europe, Earth)$$

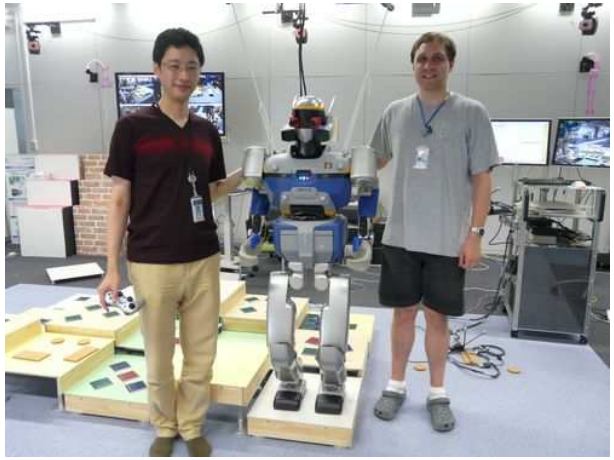
- The relation is transitive and reflexive:

$$PartOf(x, y) \wedge PartOf(y, z) \Rightarrow PartOf(x, z)$$
$$PartOf(x, x)$$

Composite objects

- Composite objects are often categorized by structural relationships between parts:

$$\begin{aligned} \mathit{Biped}(a) \quad \Rightarrow \exists\, l_1, l_2, l_3, b\ & \mathit{Leg}(l_1) \wedge \mathit{Leg}(l_2) \wedge \mathit{Body}(b) \wedge \\ & \mathit{PartOf}(l_1, a) \wedge \mathit{PartOf}(l_2, a) \wedge \mathit{PartOf}(b, a) \wedge \\ & \mathit{Attached}(l_1, b) \wedge \mathit{Attached}(l_2, b) \wedge l_1 \neq l_2 \wedge \\ & [\forall l_3\ \mathit{Leg}(l_3) \wedge \mathit{PartOf}(l_3, a) \Rightarrow (l_3 = l_1) \vee (l_3 = l_2)] \end{aligned}$$

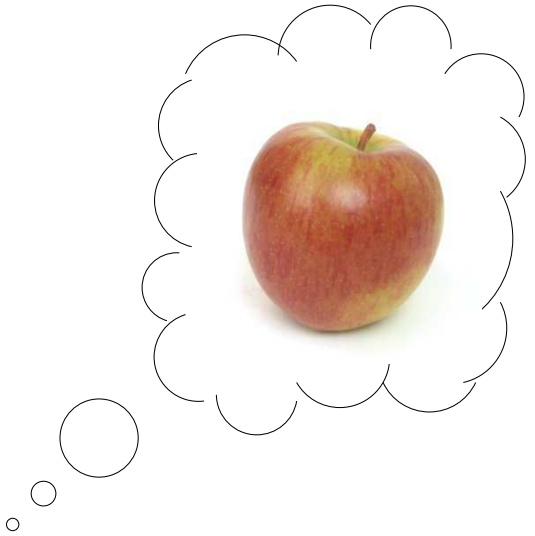


- The characterization of “has exactly two legs” is not very elegant.
- *Description logics* make this easier.

- May also want composite objects with parts and no structure.
- A bunch
$$\mathit{BunchOf}(\{\mathit{Apple}_1, \mathit{Apple}_2, \mathit{Apple}_3\}).$$
- Different from what we have seen before:
- *BunchOf*(Apples) is the composite object of all apples, distinct from *Apples*, the category of all apples.

  (The first is a concrete object, a big one, while the second is an abstract idea).

BunchOf(*Apples*)



Apples

- We can define *BunchOf* in terms of *PartOf*, since each element in the bunch is a part of it:

$$\forall x\ x \in s \ \Rightarrow\ PartOf(x, BunchOf(s))$$

- We can also say that $BunchOf(s)$ is the smallest object satsifying this condition:

$$\forall y[\forall x\ x \in s \ \Rightarrow\ PartOf(x, y)] \Rightarrow PartOf(BunchOf(s), y)$$

  $BunchOf(s)$ must be part of any object that has all the elements of s as parts.

- *Logical minimization*

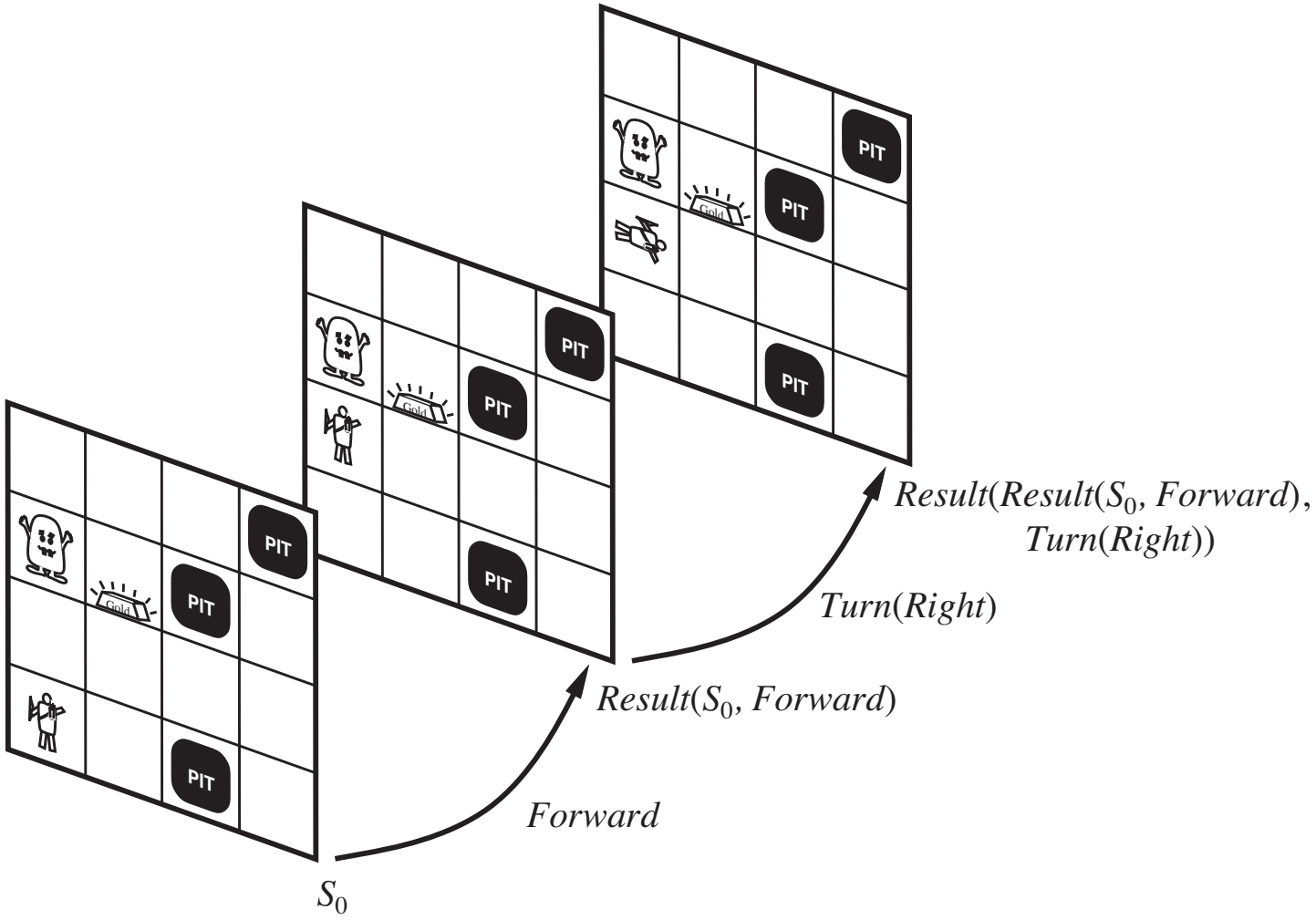
States and actions

- How do we describe the world so that we can do planning?
  - Need a way to describe states and actions.

  Remember *runs* from the first lecture?
- *Situation calculus*
- Initial state is a situation.
- s is a situation and a is an action, $Result(s, a)$ is a situation.

  A situation is a sequence (history) of actions applied to some initial state.
- Two situations are the same only if their start state and sequence of actions is the same:

$$Result(s, a) = Result(s', a') \Leftrightarrow (s = s' \wedge a = a')$$


PIT
Gold
PIT
PIT
PIT
Gold
PIT
PIT
PIT
Gold
PIT
Result(Result(S0, Forward),
Turn(Right))
Turn(Right)
Result(S0, Forward)
Forward
S0

- A *fluent* is a function or relation that can vary from one situation to another.
- By convention the situation s is the last argument to a fluent.
- $At(x, l, s)$ is the relational fluent that is true when object x is at location l in situation s.
- *Location* is a functional fluent such that $Location(x, s) = l$ when $At(x, l, s)$ is true.

- The pre-conditions of actions are described with a *possibility axiom*.
- This says when an action can be taken.
- For example:

$$Alive(Agent, s) \wedge Have(Agent, Arrow, s) \Rightarrow Poss(Shoot, s)$$

- Possibility axioms always look like:

$$\Phi(s) \Rightarrow Poss(a, s)$$

- For each fluent we need a *successor-state axiom* that says what happens to the fluent when an action is taken:

$$\begin{aligned} &Poss(a, s) \Rightarrow \\ &\quad (Holding(Agent, g, Result(a, s))) \Leftrightarrow \\ &\quad a = Grab(g) \vee (Holding(Agent, g, s) \wedge a \neq Release(g))). \end{aligned}$$

- If the action is possible, then if the fluent is true in the result state it means ($\Leftrightarrow$) the action made it true, or it was true before and the action didn't change it.
- The need to state successor axioms for each fluent/action pair is known as the *frame problem*.
- Also need axioms to say that actions with different names are different.

- Logical deduction over situation calculus descriptions gives us plans.
- That is situations (and hence sequences of actions) that satisfy a given goal.
- Of course, all of this (as written) assumes that actions are deterministic.
  - But see, for example, DTGolog.

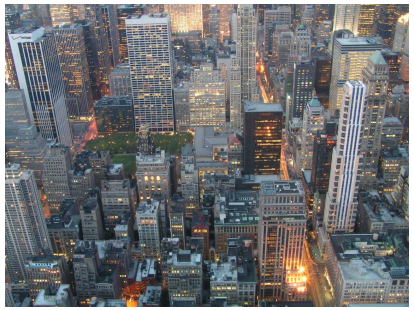
Events

- The situation calculus can describe actions and effects, but doesn't consider the duration of actions.
- Also can't handle simultaneous actions.
- So we have the *event calculus*, which is based on time points.
- "Event" is a synonym for "action".
- The event calculus reifies fluents and events.
- $At(Simon, Brooklyn)$ is an object that refers to Simon being in Brooklyn, but does not say if it is true.
- To assert its truth we say:

$$T(At(Simon, Brooklyn), this\_afternoon)$$

- Events are taken to be instances of categories of events.
- Thus the event E_1 of Simon flying from New York to London could be written as:

$$E_1 \in \mathit{Flyings} \wedge \mathit{Flyer}(E_1, \mathit{Simon}) \wedge$$
$$\mathit{Origin}(E_1, \mathit{NYC}) \wedge \mathit{Destination}(E_1, \mathit{London})$$




- If that is too long-winded, the use:

$$E_1 \in \mathit{Flyings}(\mathit{Simon}, \mathit{NY}, \mathit{London}).$$

- To express the fact that this took some time (which it does), then we can say:

$$Happens(E_1, i)$$

  where i is some time interval.

- Equivalently:

$$Extent(E_1) = i$$

- We can then identify concrete times with i by defining:

$$i = (t_1, t_2)$$

- A sample set of predicates for the event calculus is as follows.

- $T(f,t)$
  Fluent f is true at time t.
- $Happens(e,i)$
  Event e happens over interval i.
- $Initiates(e,f,t)$
  Event e causes f to start at t.
- $Terminates(e,f,t)$
  The reverse of the previous predicate.
- $Clipped(f,i)$
  f ceases to be true during i.
- $Restored(f,i)$
  The reverse of the previous predicate.

- There is a first event *Start*, and we say which fluents are initiated/terminated by *Start*.
- Then:

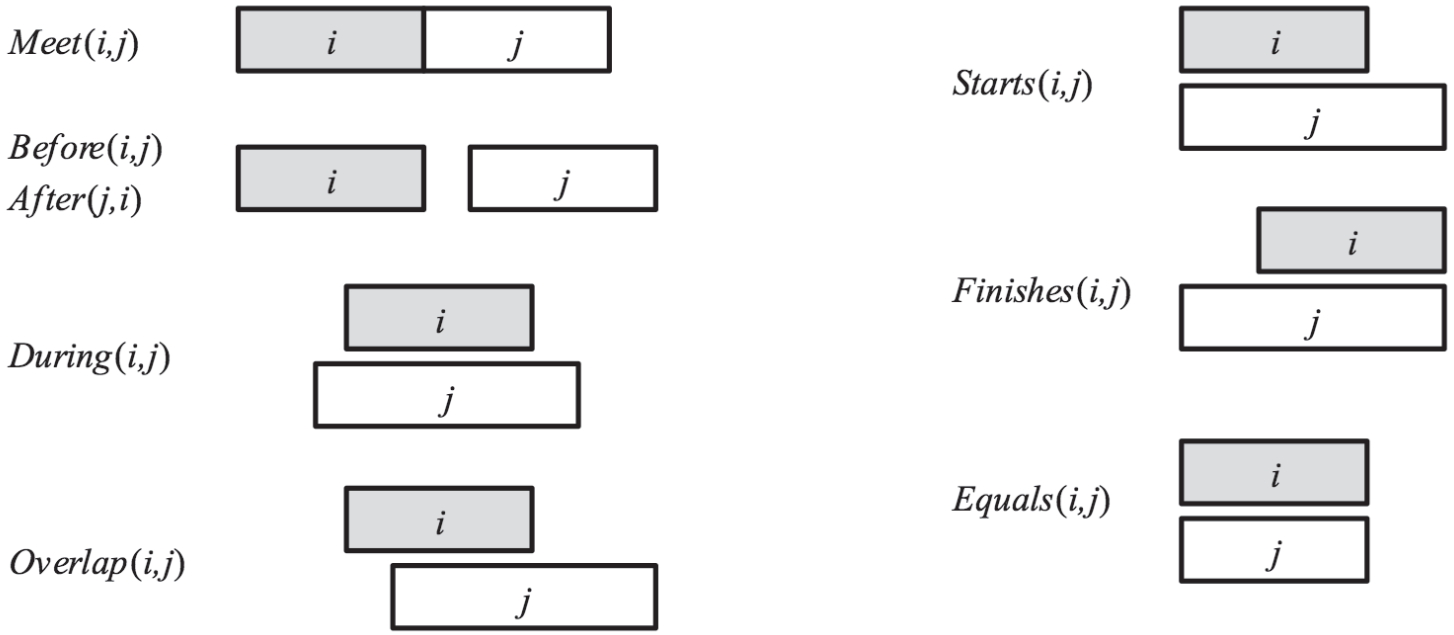
$$\mathit{Clipped}(f,(t_1,t_2)) \Leftrightarrow$$
$$\exists e,t,t_3\ \mathit{Happens}(e,(t,t_3)) \wedge t_1 \leq t \leq t_2 \wedge \mathit{Terminates}(e,f,t).$$

and

$$\mathit{Happens}(e,(t_1,t_2)) \wedge \mathit{Initiates}(e,f,t_1) \wedge \neg \mathit{Clipped}(f,(t_1,t)) \wedge t_1 < t$$
$$\Rightarrow T(f,t)$$

- And conversely for *Restored* and $\neg T$.

Temporal intervals



- All defined in terms of $Begin(i)$ and $End(i)$.

Inconsistency

- A major problem with classical logic for knowledge representation is that it does badly with inconsistency.
- Bad things happen if we have p and $\neg p$.

1.	$p \wedge \neg p$	Given
2.	p	1,$\wedge$-E
3.	$\neg p$	1,$\wedge$-E
4.	$\neg p \vee q$	2, $\vee$-I
5.	$p \Rightarrow q$	4, defn. of $\Rightarrow$
6.	q	5,$\Rightarrow$-E

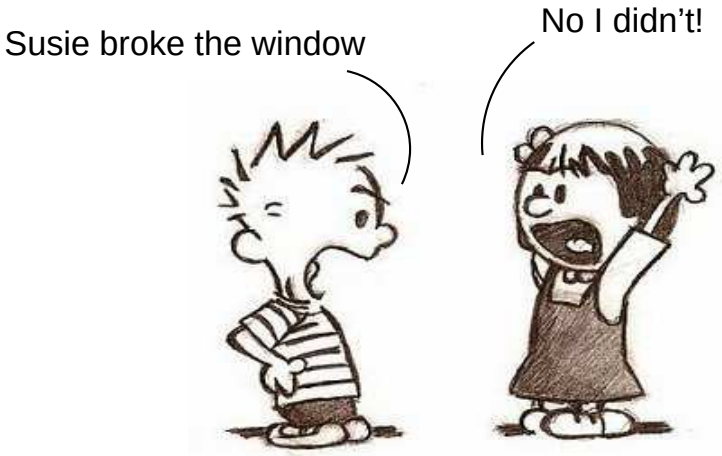
  for arbitrary (and hence any) q.

- Equally, and perhaps more explicitly, we can derive:

$$(p \wedge \neg p) \Rightarrow q$$

Why is this an issue?

- Well, one reason is that knowledge changes over time.
  - "It's raining", . . . "It's not raining", . . .
- Or we make assumptions
  - Tweety is a bird, so she must fly.
- Or we just get told things that are inconsistent:



Default logic

- Default logic is a formalism in which rules for generating assumptions can be made explicit.

$$\frac{Bird(x) : Flies(x)}{Flies(x)}$$

- If $Bird(x)$ is true, and $Flies(x)$ is consistent with the knowledge base, then conclude that $Flies(x)$.

- The general form of a default is:

$$\frac{\alpha : \beta_1, \ldots \beta_n}{\gamma}$$

  α is the *pre-requisite*, $\beta_1, \ldots \beta_n$ are the *justifications* and γ is the *conclusion*.

- When $\beta = \gamma$, the default is said to be *normal*.

- The classic default logic example, combines:

$$\frac{Bird(x) : Flies(x)}{Flies(x)}$$

with the information that

$$\{Bird(tweety)\}$$

- This allows the conclusion that $Flies(Tweety)$ since there is nothing to prevent the application of the default.

- However, pairing

$$\frac{Bird(x) : Flies(x)}{Flies(x)}$$

with the information that:

$$\{Penguin(x) \Rightarrow Bird(x), Penguin(x) \Rightarrow \neg Flies(x), Penguin(opus)\}$$

prevents the conclusion that $Flies(opus)$ since it is not consistent with $\neg Flies(opus)$ which can be inferred from the knowledge base.

- To understand what a set of default rules means, we have the notion of an *extension*
  - A maximal set of consequences.
- The above examples have only one extension. This is not always the case.
- Classic example of more than one extension is the *Nixon diamond*

- We have the defaults:

$$\left\{\frac{Quaker(x) : Pacifist(x)}{Pacifist(x)}, \quad \frac{Republican(x) : \neg Pacifist(x)}{\neg Pacifist(x)}\right\}$$

along with $\{Republican(nixon), Quaker(nixon)\}$

- We have two extensions, one including $Pacifist(nixon)$ and one including $\neg Pacifist(nixon)$.

Belief revision

- It is natural that many inferences drawn from some knowledge-base will be defaults.
- What if these turn out to be wrong?
- For example:

$$\frac{Summer(d) : Sunny(d)}{Sunny(d)}$$

  may be completely reasonable, but can still give us the wrong answer?
- Need to *revise* our beliefs.
- If we have added p to our beliefs, and then it turns out that $\neg p$ what do we do?

- One solution is just to *retract* p, remove it from the KB.
- But what if we also have:

$$p \Rightarrow q$$

  and by the time we find out that $\neg p$, we have already inferred q?
- Need to retract all the consequences of p.
- Except that maybe we also have:

$$r, r \Rightarrow q$$

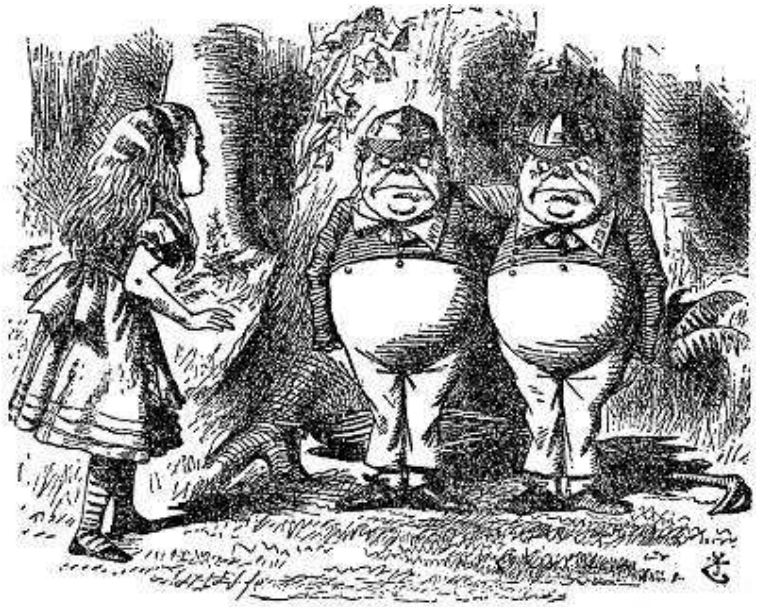
  in which case q can stay after all.
- In general, once we have $KB \vdash \perp$, we want to find the minimal S, such that $KB - S \nvdash \perp$

Truth maintenance

- Truth maintenance systems (TMS) provide one way to do this.
- In a *Justification-based truth maintenance system* (JTMS) we annotate every sentence with the set of sentences from which it was inferred.
  - A sentence can have several justifications
- In our first example q will have the justification $\{p, p \Rightarrow q\}$.
- When we want to retract p we also retract every sentence which has p in its justification.
- To cope with the $r, r \Rightarrow q$ case, we retract every s where p is in its *only* justification.

- In fact, we don't retract sentences in the sense of deleting them.
- What happens if we later find that the day is sunny (again) after all?
- Instead of retracting, we mark the sentence as being OUT, and we can swap it back to being IN if/when that is appropriate.

Argumentation



'Contrariwise,' continued Tweedledee, 'if it was so, it might be; and if it were so, it would be; but as it isn't, it ain't. That's logic.'

- Argumentation is another apporach to handling inconsistent information.

Abstract Argumentation

- We'll start by taking a step back from the details of inconsitency to an abstract notion of argument.
- Concerned with the overall structure of the set of arguments
  - (rather than internals of individual arguments).
- Write $x \rightarrow y$
  - "argument x attacks argument y";
  - "x is a counterexample of y; or
  - "x is an attacker of y".

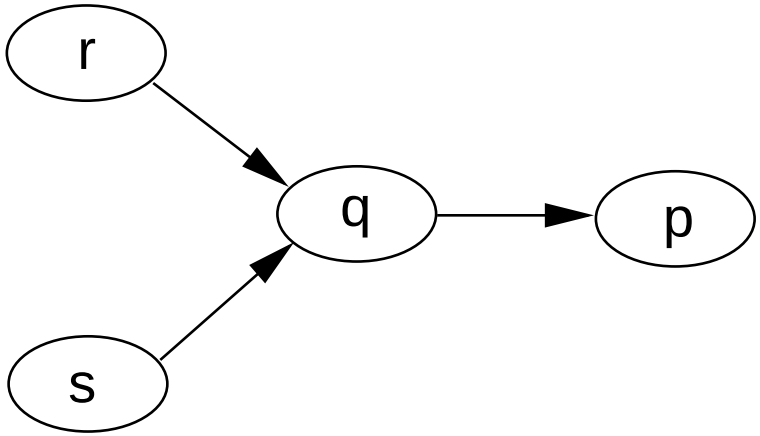
  where we are not actually concerned as to what x, y are, but the conflict between them arises from inconsistency.

- *y* might be “let’s have a picnic”, and *x* might be “it is going to rain”.
- Clearly rain is a reason to not go on a picnic.
- Later we’ll see howthe notion of “attack” can be related to statements in logic.
- An *abstract argument system* is a collection or arguments together with a relation “$\rightarrow$” saying what attacks what.

- Systems like this are called *Dung-style* after their inventor.
- A set of Dung-style arguments:

$$\langle \{p, q, r, s, \}, \{(r, q), (s, q), (q, p)\} \rangle$$

  meaning that r attacks q, s attacks q and q attacks p.



- The question is, given this, what should we believe?

Preferred extensions

- There is no universal agreement about what to believe in a given situation, rather we have a set of criteria.
- A *position* is a set of arguments.
  - Think of it as a viewpoint
- A position S is *conflict free* if no member of S attacks another member of S.
  - Internally consistent
- The conflict-free sets in the previous system are:

$$\{\}, \{p\}, \{q\}, \{r\}, \{s\}, \{r, s\}, \{p, r\}, \{p, s\}, \{r, s, p\}$$

- If an argument a is attacked by another a', then it is *defended* by a'' if a'' attacks a'.
- Thus p is defended by r and s.

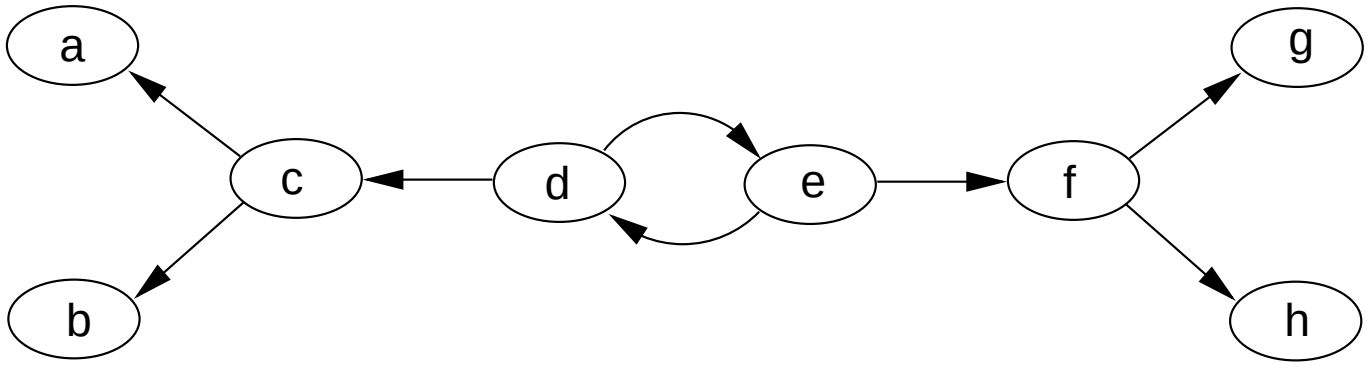
- A position S is *mutually defensive* if every element of S that is attacked is defended by some element of S.
  - Self-defence is allowed
- These positions are mutually defensive:

$$\{\}, \{r\}, \{s\}, \{r, s\}, \{p, r\}, \{p, s\}, \{r, s, p\}$$

- A position that is conflict free and mutually defensive is *admissible*.
- All the above positions are admissible.
- Admissibility is a minimal notion of a reasonable position — it is internally consistent and defends itself against all attackers.

- A *preferred extension* is a maximal admissible set.
  - adding another argument will make it inadmissible.
- In other words S is a preferred extension if S is admissible and no superset of S is admissible.
- Thus $\{\ \}$ is not a preferred extension, because $\{p\}$ is admissible.
- Similarly, $\{p, r, s\}$ is admissible because adding q would make it inadmissible.
- A set of arguments always has a preferred extension, but it may be the empty set.

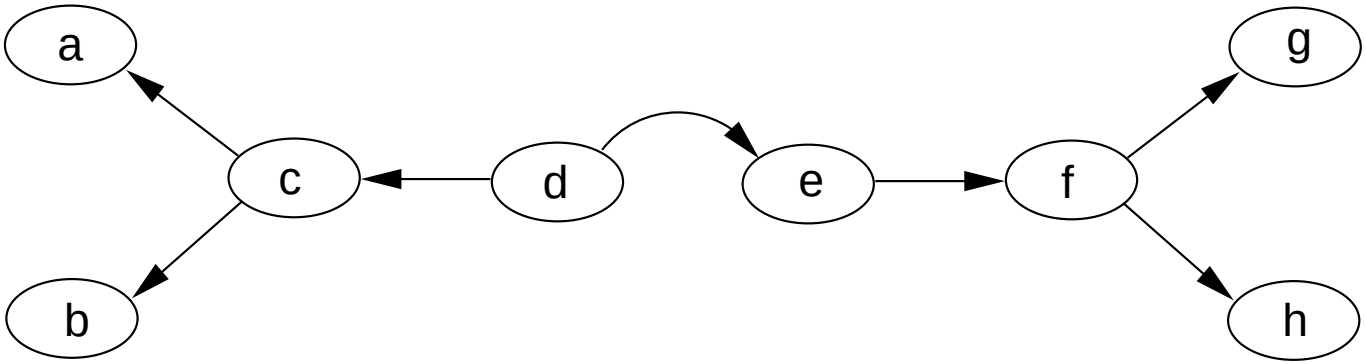
- With a larger set of arguments it is exponentially harder to find the preferred extension.
- n arguments have 2^n possible positions.
- This set of arguments:



has two preferred extensions:

$$\{a, b, d, f\} \qquad \{c, e, g, h\}$$

- In contrast:

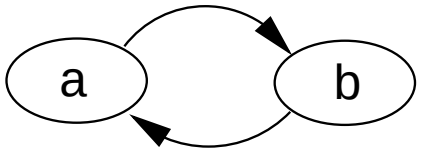


has only one:

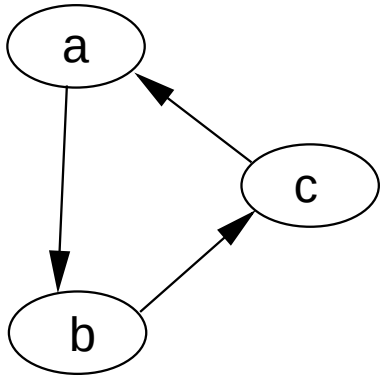
$$\{a, b, d, f\}$$

since c and e are now attacked but undefended, and so can't be in an admissible set.

- Two rather pathological cases are:



with preferred extension $\{a\}$ and $\{b\}$, and:



which has only $\{\ \}$ as a preferred extension.

Credulous and sceptical acceptance

- To improve on preferred extensions we can define

  An argument is sceptically accepted if it is a member of *every* preferred extension.

  and

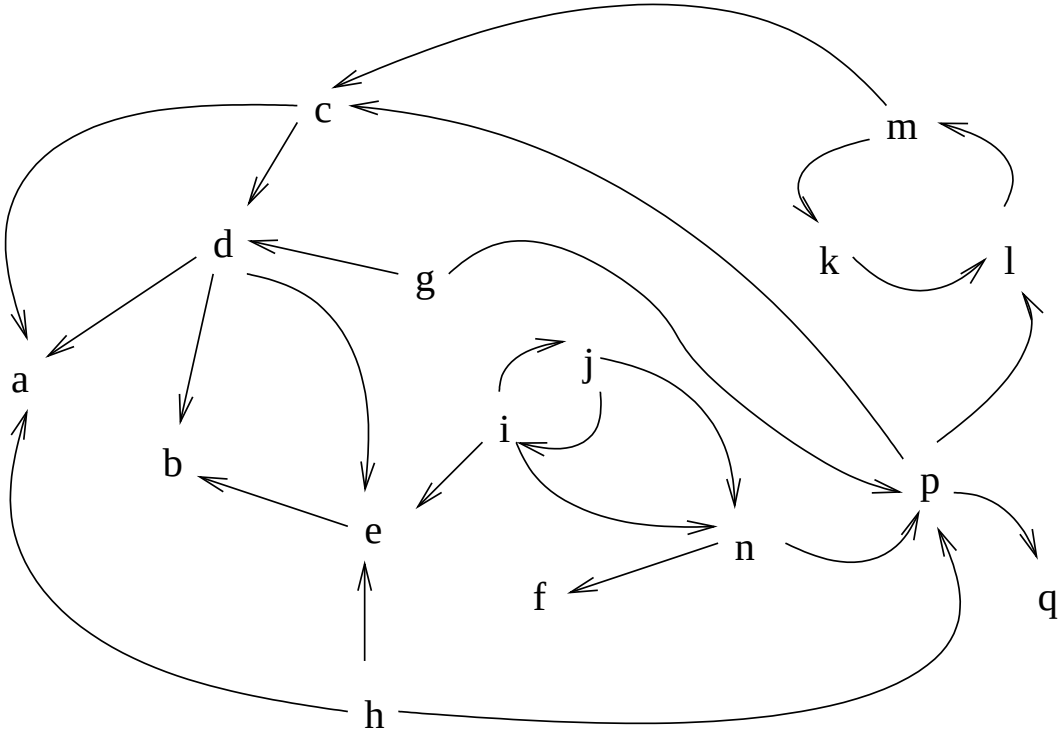
  An argument is credulously accepted if it is a member of *at least one* preferred extension.

- Clearly anything that is sceptically accepted is also credulously accepted.
- On our original example, p, q and r are all sceptically accepted, and q is neither sceptically or credulously accepted.

Grounded extensions

- Another approach, perhaps better than preferred extension.
- Arguments are guaranteed to be acceptable if they aren't attacked.
  - No reason to doubt them
- They are IN
- Once we know which these are, any arguments that they attack must be unacceptable.
- They are OUT — delete them from the graph.
- Now look again for IN arguments...
- And continue until the graph doesn't change.
- The set of IN arguments — the ones left in the graph — make up the *grounded extension*.

- Consider computing the grounded extension of:



- We can say that:
  - h is not attacked, so IN.
  - h is IN and attacks a, so a is OUT.
  - h is IN and attacks p, so p is OUT.
  - p is OUT and is the only attacker of q so q is IN.
- There is always a grounded extension, and it is always unique (though it may be empty)

Deductive Argumentation

Basic form of deductive arguments is as follows:

$$\Delta \vdash (s, G)$$

where:

- Δ is a (possibly inconsistent) set of logical formulae;
- s is a logical formula known as the *conclusion*; and
- G is a minimal consistent set of logical formulae such that:
  1. $G \subseteq \Delta$; and
  2. s can be proved from G.

Attack and Defeat

- Argumentation takes into account the relationship between arguments.
- Let (ϕ_1, Γ_1) and (ϕ_2, Γ_2) be arguments from some database Δ. Then (ϕ_2, Γ_2) can be defeated (attacked) in one of two ways:
  1. (ϕ_1, Γ_1) *rebuts* (ϕ_2, Γ_2) if $\phi_1 \equiv \neg\phi_2$.
  2. (ϕ_1, Γ_1) *undercuts* (ϕ_2, Γ_2) if $\phi_1 \equiv \neg\psi$ for some $\psi \in \Gamma_2$.
- A rebuttal or undercut is known an *attack*.
- Once we have identified attacks, we can look at preferred extensions or grounded extensions to determine what arguments to accept.

Where are we?

- Well, we've covered a lot of ground and learnt to represent a lot of information in logic.
- But we can't represent everything (sorry).
- How can we say "It may rain tomorrow".
- Or, "It is more likely than not to rain tomorrow".
- Or, "Simon is not so tall".
- Or rather, how can we say them in a way that captures the meaning that we usually place on these statements?
- Need to talk about *uncertainty* to do that.

Summary

- This lecture completes our treatment of logic.
- We looked in detail at how to represent many aspects of the world in logic.
- What we found were some solutions and more problems.
  - Many of these problems are active research topics
- Next week we'll go on to look at *uncertainty* in general and *probability* in particular.