Knowledge Representation
Project Work

Slides from Marie desJardin, David Kauchak

1

Bookkeeping

- Project designs have been returned, with feedback
  - Please feel free to talk to me about your project plan!
- Today's lecture:
  - A little more about inference
  - Knowledge Representation & Reasoning
  - Planning
    - What is planning?
    - Approaches to planning

2

Last Time: Agents

Wumpus percepts:
[Stench, Breeze, Glitter, Bump, Scream]

- Logical agents
  - Reflex: rules map directly from percepts → beliefs or percepts → actions

    $\forall$b,g,u,c,t Percept([Stench, b, g, u, c], t) → Stench(t)

    $\forall$t AtGold(t) → Action(Grab, t)
  - Model-based: construct a *model* (set of t/f beliefs about sentences) as they learn; map from models → actions

    Action(Grab, t) → HaveGold(t)

    HaveGold(t) → Action(RetraceSteps, t)
  - Goal-based: form goals, then try to accomplish them

3

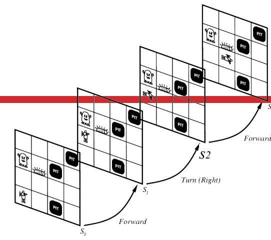
Last Time: Goal-Based Agents

- Once the gold is found, need new goals!
  - So, need a new set of actions.
- Encoded as a rule:

  ($\forall$s) Holding(Gold,s) → GoalLocation([1,1],s)
- How does the agent find a sequence of actions for goal?
- Three possible approaches are:
  - **Inference**: good versus wasteful solutions
  - **Search**: make a problem with operators and set of states
  - **Planning**: coming soon!

4

Last Time: Situations



- Representing a dynamic world
  - Situations (s_0…s_n): the world in situation 0-n
    Teaching(DrM,s_0) — today, 1:00, whenNotSick, …
  - Add 'situation' argument to statements
    AtGold(t,s_0)
  - Or, add a 'holds' predicate that says 'sentence is true in this situation'
    holds(At[2,1], s_1)
  - Or, add a result(action, situation) function that takes an action and situation, and returns a new situation
    results(Action(goNorth), s_0) → s_1

5

Last Time: Inference by Enumeration

- LET: KB = A ∨ C, B ∨ ¬C β = A ∨ B
- QUERY: KB ⊨ β ?

A	B	C	AVC	BV←C	KB	AVB	KB⇒β
false	false	false	false	true	false	false	true
false	false	true	true	false	false	false	true
false	true	false	false	true	false	true	true
false	true	true	true	true	true	true	true
true	false	false	true	true	true	true	true
true	false	true	true	false	false	true	true
true	true	false	true	true	true	true	true
true	true	true	true	true	true	true	true

β is entailed by KB if *all* models of KB are models of β, i.e., *all* rows where KB is true, β is also true

In other words: KB ⇒ β is valid

6

Automating FOL Inference with Generalized Modus Ponens

7

Automated Inference for FOL

- Automated inference using FOL is harder than PL
  - Variables can take on an infinite number of possible values
    - From their domains, anyway
    - This is a reason to do careful KR!
  - So, potentially infinite ways to apply Universal Elimination
- *Godel's Completeness Theorem* says that FOL entailment is only semidecidable*
  - If a sentence is **true** given a set of axioms, can prove it
  - If the sentence is **false**, then there is no guarantee that a procedure will ever determine this
  - **Inference may never halt**

*The "halting problem"

8

Generalized Modus Ponens (GMP)

- Apply modus ponens reasoning to generalized rules
- Combines And-Introduction, Universal-Elimination, and Modus Ponens
  - From $P(c)$ and $Q(c)$ and $(\forall x)(P(x) \wedge Q(x)) \Rightarrow R(x)$ derive $R(c)$
- General case: **Given**
  - **atomic sentences** $P_1, P_2, \ldots, P_N$
  - **implication sentence** $(Q_1 \wedge Q_2 \wedge \ldots \wedge Q_N) \Rightarrow R$
    - $Q_1, \ldots, Q_N$ and R are atomic sentences
  - **substitution** $\text{subst}(\theta, P_i) = \text{subst}(\theta, Q_i)$ for $i=1,\ldots,N$
  - **Derive new sentence: subst(θ, R)**

9

Method 3: Forward Chaining

- Proofs start with the given axioms/premises in KB, deriving new sentences using GMP until the goal/query sentence is derived
- This defines a **forward-chaining** inference procedure because it moves "forward" from the KB to the goal [eventually]
- Forward chaining with Horn clause KB is **complete**
  - A formal system is called complete with respect to a particular property if every formula having the property can be derived using that system, i.e. is one of its theorems;
  - **Intuitively, a system is called complete if it can derive every formula that is true.**

14

Forward Chaining

- "Apply" any rule whose premises are satisfied in the KB
- Add its conclusion to the KB until query is derived

KB:

$$P \Rightarrow Q$$
$$L \wedge M \Rightarrow P$$
$$B \wedge L \Rightarrow M$$
$$A \wedge P \Rightarrow L$$
$$A \wedge B \Rightarrow L$$
$$A$$
$$B$$

query: Q

15

Forward Chaining

Given KB:

1. $P \Rightarrow Q$
2. $L \wedge M \Rightarrow P$
3. $B \wedge L \Rightarrow M$
4. $A \wedge P \Rightarrow L$
5. $A \wedge B \Rightarrow L$
6. A
7. B
8. L GMP(5,6,7)
9. M GMP(3,7,8)
10. P GMP(2,8,9)
11. Q GMP(1,10)

16

Forward Chaining Exercise

- Consider the following KB:
  1. J ⇒ Q
  2. A ∧ I ⇒ J
  3. E ∧ F ⇒ I
  4. B ⇒ F
  5. A ∧ B ⇒ E
  6. A
  7. B
  8. E (GMP 5,6,7)
  9. F (GMP 4,7)
  10. I (GMP 3,8,9)
  11. J (GMP 2,6,10)
  12. Q (GMP 1,11)
- Prove Q. (Remember, you'll just use GMP over and over!)
  - **A, B, (A ∧ B) ⇒ C, ∴ C**

20

Method 4: Backward Chaining

- Forward chaining problem: can generate a lot of irrelevant conclusions
  - Search forward, start state = KB, goal test = state contains query
- Backward chaining
  - Work backwards from goal to premises
  - Find all implications of the form (…) ⇒ query
  - Prove all the premises of one of these implications
  - Avoid loops: check if new subgoal is already on the goal stack
  - Avoid repeated work: check if new subgoal
    - Has already been proved true, or
    - Has already failed

21

Backward Chaining

- Backward-chaining deduction using GMP
  - Complete for KBs containing only Horn clauses.
- Proofs:
  - Start with the goal query
  - Find rules with that conclusion
  - Prove each of the antecedents in the implication
- Keep going until you reach premises!

Avoid loops
- Is new subgoal already on goal stack?

Avoid repeated work: has subgoal
- Already been proved true?
- Already failed?

Backward Chaining Example

- KB:
  - allergies(X) ⇒ sneeze(X)
  - cat(Y) ∧ allergic-to-cats(X) ⇒ allergies(X)
  - cat(Felix)
  - allergic-to-cats(Lise)
- Goal:
  - sneeze(Lise)

Backward Chaining

Knowledge Base
1. Allergies lead to sneezing.
   allergies(X) ⇒ sneeze(X)
2. Cats cause allergies if allergic to cats.
   cat(Y) ∧ allergic-cats(X) ⇒ allergies(X)
3. Felix is a cat.
   cat(Felix)
4. Lise is allergic to cats.
   allergic-cats(Lise)

sneeze(Lise) ← query

- Backward Chaining: apply rules that end with the goal

variable binding

allergies(X) → sneeze(X) + sneeze(Lise)
new query: allergies(Lise)?

cat(Y) ∧ allergic-cats(X) → allergies(X) + allergies(Lise)
new query: cat(Y) ∧ allergic-cats(Lise)?

cat(Felix) + cat(Y) ∧ allergic-cats(Lise)
new sentence: cat(Felix) ∧ allergic-cats(Lise) ✓

25

Forward vs. Backward Chaining

- FC is data-driven
  - Automatic, unconscious processing
  - E.g., object recognition, routine decisions
  - May do lots of work that is irrelevant to the goal
- BC is goal-driven, appropriate for problem-solving
  - Where are my keys? How do I get to my next class?
  - Complexity of BC can be much less than linear in the size of the KB

27

Completeness of GMP

- GMP (using forward or backward chaining) is complete for KBs that contain only Horn clauses
- It is ***not* complete** for simple KBs that contain non-Horn clauses
- The following KB entails that S(A) is true:
  - $(\forall x)\ P(x) \Rightarrow Q(x)$
  - $(\forall x)\ \neg P(x) \Rightarrow R(x)$
  - $(\forall x)\ Q(x) \Rightarrow S(x)$
  - $(\forall x)\ R(x) \Rightarrow S(x)$
- If we want to conclude S(A), with GMP we cannot, since the second one is not a Horn clause
- It is equivalent to $P(x) \vee R(x)$

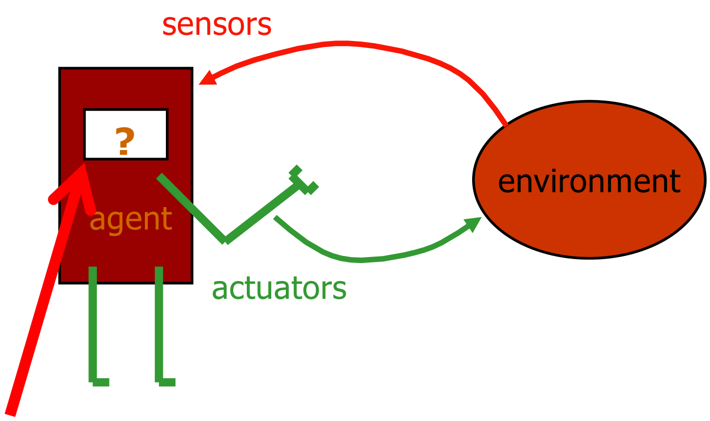
28

Knowledge Representation and Reasoning (KR&R)

Chapters 12.1-12.2, 12.5-12.6

50

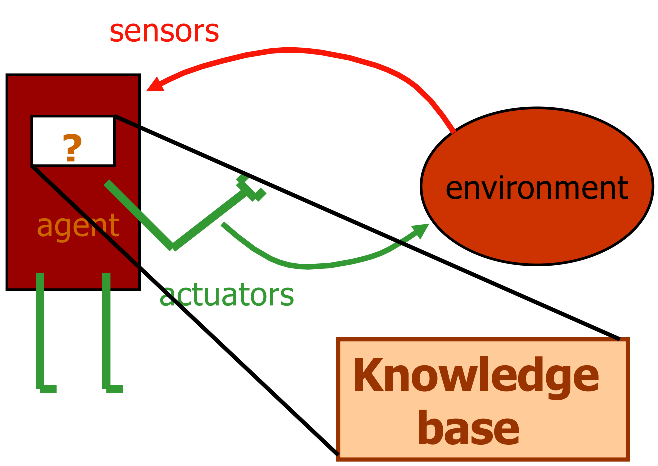
Agent's knowledge representation



What have we seen so far for knowledge representation?

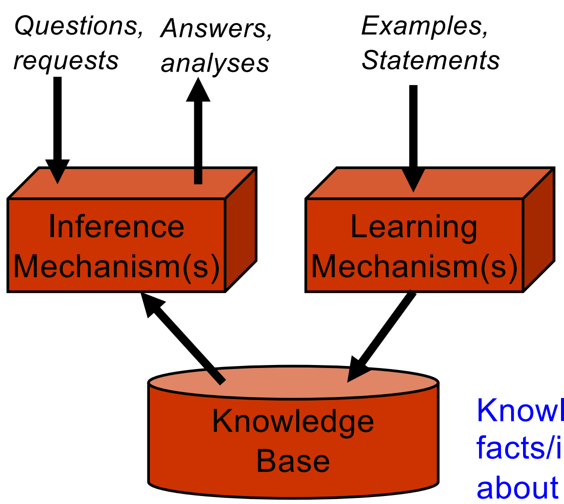
51

Knowledge-based agent



52

Knowledge-based approach



Knowledge base stores facts/information/rules about the world

53

What is in a knowledge base?

- Facts…
- Specific:
  - Middlebury College is a private college
  - Prof. Kauchak teaches at Middlebury College
  - 2+2 = 4
  - The answer to the ultimate question of life is 42
- General:
  - All triangles have three sides
  - All tomatoes are red
  - n2 = n * n

54

Inference

- Given facts, we'd like to ask questions
  - Key: depending on how we store the facts, this can be easy or hard
  - People do this naturally (though not perfectly)
    - For computers, we need specific rules
- For example:
  - Johnny likes to program in C
  - C is a hard programming language
  - Computer scientists like to program in hard languages
- What can we infer?

55

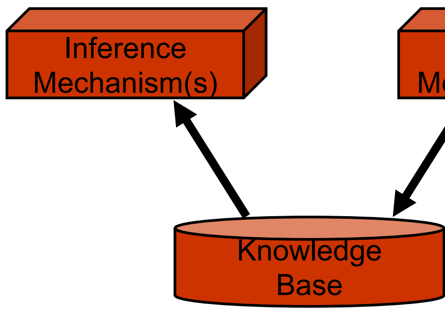
Inference

- For example:
  - Johnny likes to program in C
  - C is a hard programming language
  - Computer scientists like to program in hard languages
- Be careful!
  - we cannot infer that Johnny is a computer scientist
- What about now:
  - All people who like to program in hard languages are computer scientists
- What can we infer?

56

Creating a knowledge-based agent

- **Representation: how are we going to store our facts?**
- Inference: How can we infer information from our facts? How can we ask questions?
- Learning: How will we populate our facts?



57

Introduction

- Real knowledge representation and reasoning systems: several varieties
- These differ in their intended use, expressivity, features,…
- Some major families are
  - Logic programming languages
  - Theorem provers
  - Rule-based or production systems
  - Semantic networks
  - Frame-based representation languages
  - Databases (deductive, relational, object-oriented, etc.)
  - Constraint reasoning systems
  - Description logics
  - Bayesian networks
  - Evidential reasoning

58

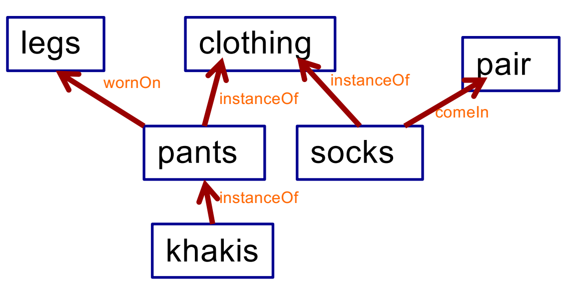
Ontologies

- Representations of general concepts
- Usually represented as a type hierarchy
  - Sort of a special case of a semantic network (wait for it...)
- "Ontological engineering" is hard!
  - How do you create an ontology for a particular application?
  - How do you maintain an ontology for changing needs?
  - How do you merge ontologies from different fields?
  - How do you map across ontologies from different fields?

59

Ontology

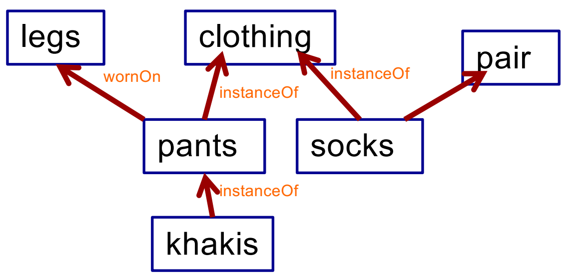
- First-order logic states relationships between objects
- One easy way to represent a similar concept is with a graph
  - nodes are the objects
  - edges represent relationships between nodes
  - some of the quantifier capability is lost



60

Ontology

- Intuitive representation for people
- Can pose questions as graph traversals which is often more comfortable/efficient



61

Upper Ontologies

- Highest-level categories: typically these might include:
  - Measurements
  - Objects and their properties (including fluent, or changing, properties)
  - Events and temporal relationships
  - Continuous processes
  - Mental events, processes; "beliefs, desires, and intentions"
- Also useful:
  - Subtype relationships
  - PartOf relationships
  - Composite objects

62

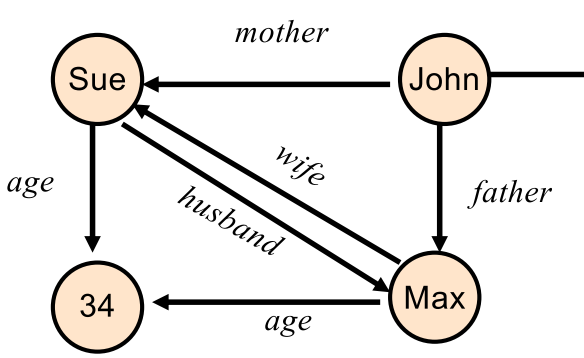
Semantic Networks

- A semantic network is a representation scheme that uses a graph of **labeled nodes** and **labeled, directed arcs** to encode knowledge.
  - Usually used to represent static, taxonomic, concept dictionaries
- Typically used with a special set of procedures to perform reasoning
  - e.g., inheritance of values and relationships
- Semantic networks were very popular in the ‘60s and ‘70s but are less frequently used today.
  - Often much less expressive than other KR formalisms
- The **graphical depiction** associated with a semantic network is a significant reason for their popularity.

63

Nodes and Arcs

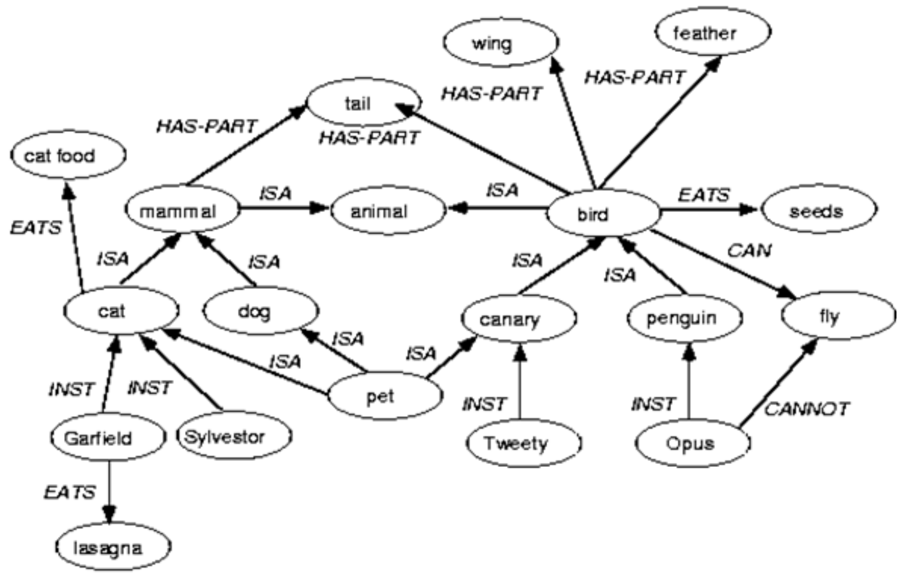
- **Arcs** define binary relationships that hold between objects denoted by the **nodes**.



mother(john,sue)
age(john,5)
wife(sue,max)
age(max,34)
...

64

Semantic Network Example

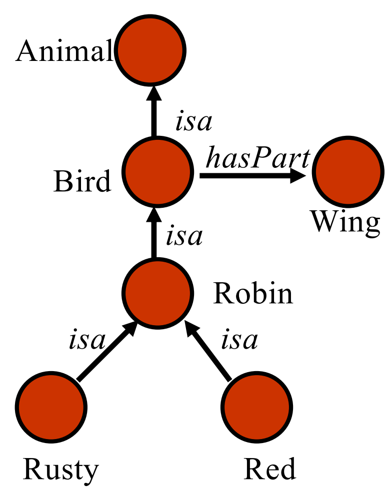


Source: http://zeus.csci.unt.edu/swigger/csci3210/

65

Semantic Networks

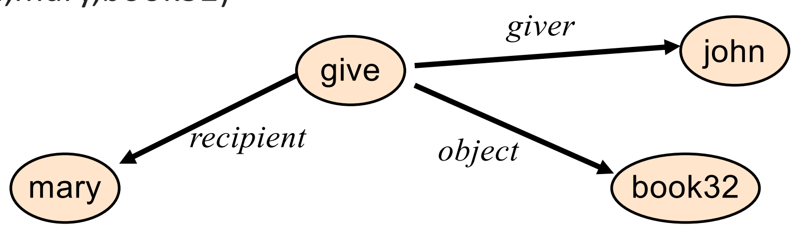
- The ISA (is-a) or AKO (a-kind-of) relation is often used to link instances to classes, classes to superclasses
- Some links (e.g. hasPart) are inherited along ISA paths.
- The semantics of a semantic net can be informal or very formal
  - often defined at the implementation level



66

Reification

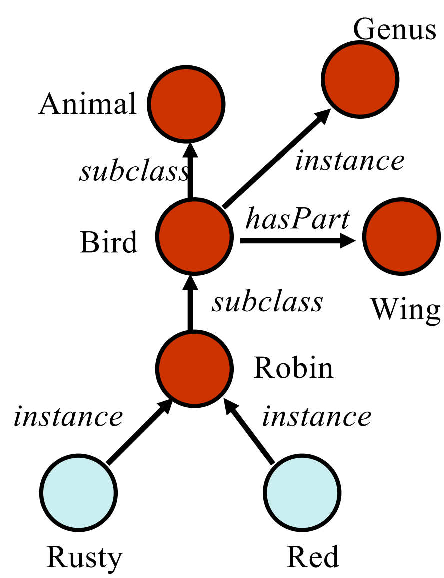
- Non-binary relationships can be represented by “turning the relationship into an object”
- This is an example of what logicians call “reification”
- We might want to represent the generic give event as a relation involving three things: a giver, a recipient and an object, give(john,mary,book32)



67

Individuals and Classes

- Many semantic networks distinguish:
  - Nodes representing individuals and those representing classes
  - The “subclass” relation from the “instance-of” relation



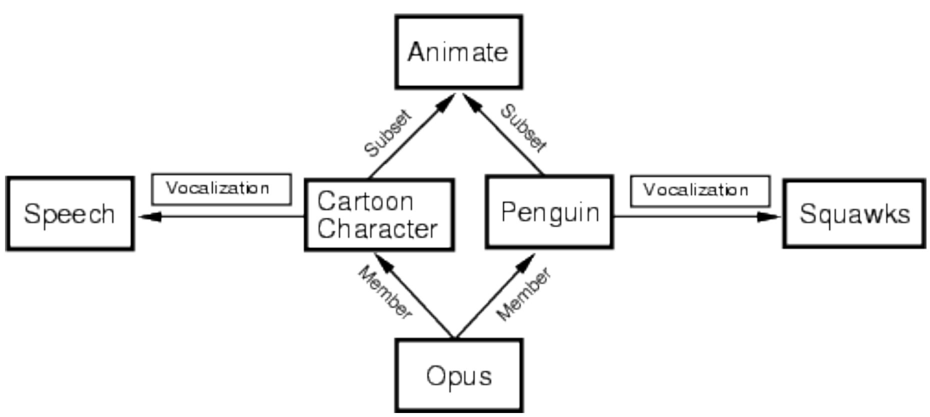
68

Inference by Inheritance

- One of the main kinds of reasoning done in a semantic net is the inheritance of values along the subclass and instance links.
- Semantic networks differ in how they handle the case of inheriting multiple different values.
  - All possible values are inherited, or
  - Only the "lowest" value or values are inherited

70

Conflicting Inherited Values



71

Multiple Inheritance

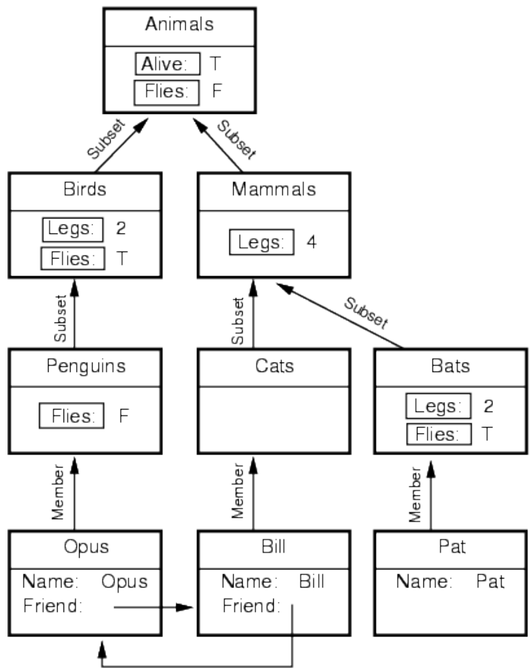
- A node can have any number of superclasses that contain it, enabling a node to inherit properties from multiple “parent” nodes and their ancestors in the network.
- These rules are often used to determine inheritance in such “tangled” networks where multiple inheritance is allowed:
  - If X<A<B and both A and B have property P, then X inherits A’s property.
  - If X<A and X<B but neither A<B nor B<A, and A and B have property P with different and inconsistent values, then X does not inherit property P at all.

72

From Semantic Nets to Frames

- Semantic networks morphed into Frame Representation Languages in the ‘70s and ‘80s
- A frame is a lot like the notion of an object in OOP, but has more meta-data
- A **frame** has a set of **slots**.
- A **slot** represents a relation to another frame (or value).
- A slot has one or more **facets**.
- A **facet** represents some aspect of the relation.

74



Rel(Alive,Animals,T)
Rel(Flies,Animals,F)

Birds ⊂ Animals
Mammals ⊂ Animals

Rel(Flies,Birds,T)
Rel(Legs,Birds,2)
Rel(Legs,Mammals,4)

Penguins ⊂ Birds
Cats ⊂ Mammals
Bats ⊂ Mammals

Rel(Flies,Penguins,F)
Rel(Legs,Bats,2)
Rel(Flies,Bats,T)

Opus ∈ Penguins
Bill ∈ Cats
Pat ∈ Bats

Name(Opus,"Opus")
Name(Bill,"Bill")
Friend(Opus,Bill)
Friend(Bill,Opus)
Name(Pat,"Pat")

(a) A frame–based knowledge base **(b) Translation into first–order logic**

75

Description Logics

- Description logics provide a family of frame-like KR systems with a formal semantics.
  - E.g., KL-ONE, LOOM, Classic, …
  - These logics can be used to determine whether categories belong within other categories (i.e., subsumption tasks)
- An additional kind of inference done by these systems is automatic classification
  - finding the right place in a hierarchy of objects for a new description
- Current systems take care to keep the languages simple, so that all inference can be done in polynomial time (in the number of objects)
  - ensuring tractability of inference

76

Abduction

- Abduction is a reasoning process that tries to form plausible explanations for abnormal observations
  - Abduction is distinctly different from deduction and induction
  - Abduction is inherently uncertain
- Uncertainty is an important issue in abductive reasoning
- Some major formalisms for representing and reasoning about uncertainty:
  - Mycin's certainty factors (an early representative)
  - **Probability theory (esp. Bayesian belief networks)**
  - Dempster-Shafer theory
  - Fuzzy logic
  - Truth maintenance systems
  - Nonmonotonic reasoning

77

Abduction

- Definition (Encyclopedia Britannica): reasoning that derives an explanatory hypothesis from a given set of facts
  - The inference result is a **hypothesis** that, **if true**, could explain the occurrence of the given facts
- Examples
  - Dendral, an expert system to construct 3D structure of chemical compounds
    - Fact: mass spectrometer data of the compound and its chemical formula
    - KB: chemistry, esp. strength of different types of bounds
    - Reasoning: form a hypothetical 3D structure that satisfies the chemical formula, and that would most likely produce the given mass spectrum

78

Abduction Examples (cont.)

- Medical diagnosis
  - Facts: symptoms, lab test results, and other observed findings (called manifestations)
  - KB: causal associations between diseases and manifestations
  - Reasoning: one or more diseases whose presence would causally explain the occurrence of the given manifestations
- Many other reasoning processes (e.g., word sense disambiguation in natural language process, image understanding, criminal investigation) can also been seen as abductive reasoning

79

Abduction, Deduction, Induction

Deduction:	major premise:	All balls in the box are black	A => B A --------- B
	minor premise:	These balls are from the box	
	conclusion:	These balls are black	
Abduction:	rule:	All balls in the box are black	A => B B ------------- Possibly A
	observation:	These balls are black	
	explanation:	These balls are from the box	
Induction:	case:	These balls are from the box	Whenever A then B ------------- Possibly A => B
	observation:	These balls are black	
	hypothesized rule:	All ball in the box are black	

Deduction reasons from causes to effects
Abduction reasons from effects to causes
Induction reasons from specific cases to general rules

80

Characteristics of Abductive Reasoning

- "Conclusions" are hypotheses, not theorems (may be false even if rules and facts are true)
  - E.g., misdiagnosis in medicine
- There may be multiple plausible hypotheses
  - Given rules A => B and C => B, and fact B, both A and C are plausible hypotheses
  - Abduction is inherently uncertain
  - Hypotheses can be ranked by their plausibility (if it can be determined)

81

Characteristics of Abductive Reasoning (cont.)

- Reasoning is often a hypothesize-and-test cycle
  - **Hypothesize**: Postulate possible hypotheses, any of which would explain the given facts (or at least most of the important facts)
  - **Test**: Test the plausibility of all or some of these hypotheses
  - One way to test a hypothesis H is to ask whether something that is currently unknown–but can be predicted from H–is actually true
    - If we also know A => D and C => E, then ask if D and E are true
    - If D is true and E is false, then hypothesis A becomes more plausible (**support** for A is increased; **support** for C is decreased)

82

Characteristics of Abductive Reasoning (cont.)

- Reasoning is **non-monotonic**
  - That is, the plausibility of hypotheses can increase/decrease as new facts are collected
  - In contrast, deductive inference is monotonic: it never change a sentence's truth value, once known
  - In abductive (and inductive) reasoning, some hypotheses may be discarded, and new ones formed, when new observations are made

83