

Computer Science 348

Introduction to Artificial Intelligence

Spring 2004: Quiz #2

Read each question carefully. **Think** before you answer. Vague answers will get few points.
Good luck!

1. Help implement CAESAR in the Deductive Retriever. Assume that facts will be asserted of the form (course name (time day start stop) teacher room), e.g., (course cs348 (time tue 9.5 11) riesbeck m128), and (between ?n ?n1 ?n2) has been defined to implement “the number ?n is between ?n1 and ?n2.” Start and stop are decimal military times, e.g., 2:30 is 14.5. Note any restrictions on queries that your rules impose.

a. Implement (teaches ?t ?c) and (location ?c ?r) to mean ?t teaches course ?c, and course ?c meets at room ?r, respectively.

```
(tell '(<- (teaches ?t ?c) (course ?c ?time ?t ?r))
(tell '(<- (location ?c ?r) (course ?c ?time ?teacher ?r)))
```

- *No need for (time ...) functional term here*

b. Implement (occupied ?r ?t) to mean room ?r is busy with a class at time ?t.

```
(tell '(<- (occupied ?r (time ?d ?s1 ?e1))
          (course ?c (time ?d ?s2 ?e2))
          (overlaps ?s1 ?e1 ?s2 ?e2)))

(tell '(<- (overlaps ?s1 ?e1 ?s2 ?e2) (between ?s1 ?s2 ?e2)))
(tell '(<- (overlaps ?s1 ?e1 ?s2 ?e2) (between ?s2 ?s1 ?e1)))
```

1. *Need to use (time ...) functional term in rule because a bare number doesn't give enough information.*
2. *Need two rules (but no more) rules to catch possible overlaps, whether you define a separate overlaps predicate or not.*

c. Implement (conflicts ?c1 ?c2) to mean course ?c1 has a time conflict with course ?c2.

```
(tell '(<- (conflicts ?c1 ?c2)
          (course ?c1 (time ?d ?s1 ?e1) ?t1 ?r1)
          (course ?c2 (time ?d ?s2 ?e2) ?t2 ?r2)
          (overlaps ?s1 ?e1 ?s2 ?e2)))
```

- *Need different teacher and room variables for two courses but same day variable.*
- *Do not need a “same” predicate for day.*
- *If overlaps not defined previously, need two rules to catch both overlap situations.*

2. Represent the following using first-order logic if possible. If not possible, say why not and represent with higher-order logic.

- *Except for 2a, all of these are major challenges to represent in logic. I was looking for the gist of the basic concepts that needed to be captured, not complete forms. The major mistakes was writing English with parentheses, e.g., $\text{win}(x)$ or $\text{wouldHaveDo}(x, y)$, which are so vague they say nothing.*

a. Everybody loves somebody sometime.

$$\forall x \in \text{People}, \exists y \in \text{People}, t \in \text{time} \text{ loves}(x, y, t)$$

- *Need to say this is about People, not all x and y .*
- *Need time – time intervals probably most appropriate..*

b. You can't win them all.

$$\forall x \in \text{People}, \exists e \in \text{EventSequence} \text{ outcome}(e) \neq \text{desiredOutcome}(x, e)$$

- *not $\text{win}(x)$ is way too vague about what x is, what winning means. Ditto referring to a “winnable” event.*
- *Talking about competition is closer but doesn't cover all cases where this is said.*
- *Critical features are some sort of outcome (causal sequence) that could've turned out differently, and desirability for agent of outcome. Explicitly representing “could have happened” would require more advanced logic, e.g., modal logic.*
- *There is no “you” constant in logic. In this context, “you” means everyone. In dialog, “you” is a way to refer to the listener, but it's like “this” – not a fixed constant object.*

c. Lightning never strikes twice in the same place.

$$\forall l_1, l_2 \in \text{LightningEvent} \ l_1 \neq l_2 \Rightarrow \text{location}(l_1) \neq \text{location}(l_2)$$

- *The above was accepted, but this proverb isn't really just about lightning. It's about any rare event. But capturing the notion of rare events not reoccurring in some “similar” way is way beyond FOL and very hard to pin down.*
- *Common mistakes in doing the logic were saying the equivalent of all lightning occurs in places (but not distinguished) or that there exists lightning in some other place, or not being clear that lightning is an event.*

d. Do unto others as you would have them do unto you.

$$\forall x \in \text{People} \ \exists y, a \ y \in \text{People}, a \in \text{actions} \ \text{wants}(x, \text{do}(y, a, x)) \Rightarrow \text{correct}(\text{do}(x, a, y))$$

- *The sentence is an imperative. Imperatives are neither true or false. The logical equivalent requires a predicate like “should do” or “is correct behavior.”*
- *A $\text{do}()$ predicate could mean either “actor did action” which isn't appropriate here where no actual doing has occurred, or “actor commonly does action” which might be OK but not all you need here.*
- *The above uses a do function term to represent the event category (see textbook) for doing of an action. Both predicates, wants and correct are about event categories here.*
- *Key ideas captured are connection between wanting something and the “correctness” of doing it yourself.*
- *Again, there is no “you” constant.*

3. Represent our everyday understanding of buying something – not shopping, just the actual purchase -- using first-order logic. Document the predicates and functions you use in the spaces provided. Note any facts about buying you can't represent.

Predicates: List each predicate used in your rules below, with a brief descriptive phrase.

```
buys(x, y, o, t) - x buys o from y at time t
transfer(x, y, o, t) - x transfers ownership of o to y at time t
owns(x, o, i) - x owns o during interval i
agent(x) - x is an agent (person, company, ...)
object(x) - x is a ownable thing (physical object, money)
time(t) - t is a point in time
interval(i) - i is a time interval
```

Functions: List each function used in your rules below, with a brief descriptive phrase.

```
beginPoint(i) - the start time of a time interval
endPoint(t) - the end time of a time interval
```

Logic: Put your facts and implications here.

```
for all x, y, o, t: agent(x), agent(y), object(o), time(t)
buys(x, o, y, t) =>
  transfer(x, o, y, t) & transfer(y, price(o), x, t)
```

```
for all x, y, o, t: agent(x), agent(y), object(o), time(t)
transfer(x, o, y, t) =>
  exists before, after: interval(before), interval(after)
  & endpoint(before) = t & beginPoint(after) = t
  & owns(x, o, before)
  & not owns(x, o, after)
  & owns(y, o, after)
```

- *The core concepts in buying something is the simultaneous transfer of ownership of an object and money equal to the price of the object.*
- *This has to involve time or situations, otherwise inconsistent predications will be made.*
- *You can't say that the buyer owns the object for all times after the purchase. Buyer could see it or lose it. How long ownership lasts is unknown.*
- *Same thing for saying seller owned object for all times before the purchase.*
- *Using two transfers allows the intervals for owning the object and the money to be different.*
- *Not captured here is the causality – this just says buying means these two transfers occurred but there's no causal link, no explicit representation of obligation.*
- *Nor can the above handle buying a service, because a service can't be owned and transferred in the above sense.*
- *Several people wrote a logic program here, not first order logic. Doesn't work well at all.*