CS262 Info Mgmt Blaheta

Homework 1

Due: 13 Sep 2007

Problem 1.1

J.R.R. Tolkien wrote a poem that has a lot of quantified statements in it:

> All that is gold does not glitter,
>   Not all those who wander are lost;
> The old that is strong does not wither,
>   Deep roots are not reached by the frost.   —J.R.R. Tolkien

a. Using O_x "x is old", S_x "x is strong", and W_x "x withers", you could translate the third line symbolically as

$$\forall_x \left[O_x \underline{\quad} S_x \to \neg W_x \right]$$

(SEE FOOTNOTE[1]) What operator would fill in the blank? Why? How do the operations associate (i.e. where should the parentheses go)?

b. For the second line, the "not" means there are two ways to translate the proposition: one more literal, using a $\forall$, and one that means basically the same thing, using $\exists$. (Think about this for a moment.) Are there any circumstances where they don't mean precisely the same thing? What are they, or why not?

c. Write out a direct translation of the first line into first-order predicate logic, along with a dictionary to interpret your symbols. Then step back, reread the line (and the rest of the poem) and write out symbolically what it ought to mean, and a translation back into English. What's going on here?

Problem 1.2

In most languages you've seen, to store the age of a person we might have an instance variable `age`, so that `sam.age` contains an integer, or perhaps a

[1] As originally published, this formula was missing the "$\neg$". Sorry about that!

function, so that `age(sam)` returns an integer (and in both cases, `sam` is a variable containing an object or structure).

In Prolog, though, you've only seen how to assert things as relationships between identifiers. You can also assert relationships between identifiers (like `sam`) and numbers, so in addition to `mother` and `father`, you can have

```
age (alex, 6).
age (chris, 8).
age (jordan, 9).
age (loren, 5).
```

Write a predicate `older/2` (that is, one named `older` that is binary, taking two arguments) that makes use of this information.

Problem 1.3

Build on the definition of `older` by defining three predicates `oldest_child/1`, `youngest_child/1`, and `middle_child/1` that are true if and only if their argument is, respectively, the oldest, the youngest, or a middle child among a set of siblings. Someone who is an only child is both oldest and youngest, but not a middle child.

You can (and will need to) assume in writing these rules that ages have been entered for everyone, and thus that `older` is defined.

Hand in the Prolog rules as assignment `hwk1`.