6.824 Artificial Intelligence, Fall 1993
Problem Set 10

Due: Beginning of class Friday, December 3

Problem 1. A *combinator* is a closed term of higher order logic. Intuitively, a combinator is a term whose meaning is fixed — the meaning of a combinator does not depend on the meaning of free variables. Below is a list of specifications for higher order combinators. For each specification give a term of higher order logic satisfying the specification, i.e., implement the desired function in higher order logic.

$$\texttt{SUBREL} : (D \times D \to B) \times (D \times D \to B) \to B$$

The assertion (SUBREL R W) should be true if R is a subrelation of W, i.e., if (W x y) is true for all x and y such that (R x y) is true.

$$\texttt{REL-COMP} : (D \times D \to B) \times (D \times D \to B) \to (D \times D \to B)$$

The expression (REL-COMP R W) is the relation that is the composition of R and W. More formally, ((REL-COMP R W) x y) should be true if there is some z such that (R x z) is true and (W z y) is true. For example, the composition of the parent relation with itself is the grandparent relation and the composition of the parent relation with the gradparent relation is the great-grandparent relation.

$$\texttt{TRANS} : (D \times D \to B) \to B$$

The expression (TRANS R) should be true if R is transitive. Your implementation should be of the form (λ (R) $\Phi[R]$) where $\Phi[R]$ does not contain any λ-expressions (or quantifiers which have implicit λ-expressions) other than the combinators SUBREL and REL-COMP.

$$\texttt{REL-UNION} : (D \times D \to B) \times (D \times D \to B) \to (D \times D \to B)$$

The expression (REL-UNION R W) is the relation that is the relation union of R and W. More formally, ((REL-UNION R W) x y) should be true if either (R x y) is true or (W x y) is true.

$$\texttt{KLEENE-STAR} : (D \times D \to B) \to (D \times D \to B)$$

The expression (KLEENE-STAR R) the infinite union of the identity relation and any number of compositions of R with itself. This is equivalent to saying that

(KLEENE-STAR R) is the least transitive relation containing the identity relation and R. Hint: The answer is a λ-expression of the form

$$(\lambda\ (r)\ (\lambda\ (x\ y)\ \exists\ r'\ \text{“}r' = r^* \text{ and } (r'\ x\ y)\text{”}))$$

$$\texttt{APPLY} : (D \times D \to B) \times (D \to B) \to (D \to B)$$

The expression (APPLY R S) should denote the "application" of the relation R to the set S. More formally (APPLY R S) should denote the set of individuals y such that there exits an individual x such that (S x) and (R x y) are both true.

$$\texttt{EVERY} : (D \to B) \to (D \to B) \to B$$

The assertion ((EVERY P) Q) should be true if P is a subset of Q. More formally, ((EVERY P) Q) should be true if (Q x) is true for every x such that (P x) is true.

$$\texttt{SOME} : (D \to B) \to (D \to B) \to B$$

The assertion ((SOME P) Q) should be true if the set P intersects the set Q. More formally, ((SOME P) Q) should be true if there is some x such that (Q x) and (P x) are both true.

Problem 2. We define a "quantifier" to be an object of type $(D \to B) \to B$. For example, (EVERY BIRD) is a quantifier. Modify the combinator APPLY so that it takes a quantifier as an argument and such that (APPLY R (SOME P)) is the set of y such that there exists an x such that (P x) and (R x y) and (APPLY R (EVERY P)) is the set of y such that for all x such that (P x) we have (R x y). Your solution should have the property that ((SOME WOMAN) (APPLY LIKES (EVERY MAN))) has a meaning that is one of the intuitive meanings of the sentence "some woman likes every man".

Problem 3. In this problem we consider computing (as opposed to defining) the values of expressions in higher order logic.

Part a. Consider the following definition of a function V which computes a "value" for an expression in higher order logic. Fill in the missing parts of the definition. We will only be concerned with expressions that have no free variables and all of whose constants are of types D, $D \to B$ or $D \times D \to B$ and whose only quantifiers, i.e., expressions of the form $\forall_\tau$ or $\exists_\tau$, only quantify over D, i.e., are of the form $\forall_D$ and $\exists_D$. We assume that each constant c has an interpretation denoted by $M(c)$. We also assume that each element of D is named by some constant of thype D and that there are only finitely many elements of the set denoted by D. If f is a constants of type

$D \to B$ and $D \times D \to B$ then $M(f)$ is an array of Boolean values indexed by the elements of D. Your completion of the following definition should have the property that if P is an expression of type B satisfying these conditions then the equations provide a way of computing $V[P]$ as a truth value.

- $V[c] = M(c)$ where c is a variable of type D.
- $V[(f\ t)] = M(f)[V[t]]$ where f is a constant of type $D \to B$ and t is a term of type D.
- $V[(f\ t\ w)] = M(f)[V[t],\ V[w]]$ where f is a constant of type $D \times D \to B$ and t is a term of type D.
- $V[(\lambda\ (x_1 \ldots x_n)\ b[x_1\ \ldots\ x_n])] = (\lambda\ (x_1 \ldots x_n)\ b[x_1\ \ldots\ x_n])$
- $V[((\lambda\ (x_1 \ldots x_n)\ b[x_1\ \ldots\ x_n])\ u_1 \ldots u_n)] = ?$
- $V[(\forall_D\ (\lambda\ (x)\ \Phi[x]))] = ?$
- $V[(\exists_D\ (\lambda\ (x)\ \Phi[x]))] = ?$
- $V[((s\ w_1 \ldots w_m)\ u_1 \ldots u_n)] = ?$

Part b. Give the order of running time as a function of n (the number of elements in the interpretation of D) used to compute the value of the following expressions using the recursive evaluator defined in part a. Remember that each combinator is just an abbreviation for a closed expression of higher order logic. Explain your answers.

- `((EVERY P) Q)`
- `((SOME P) Q)`
- `((EVERY P) (APPLY R (SOME Q)))`
- `((SOME P) (APPLY R (EVERY (APPLY W (SOME Q)))))`
- `((EVERY (APPLY R (SOME P))) Q)`

Problem 4. Direct evaluation of the sort described in problem 2 is not possible for infinite or unbounded domains. In this problem we consider inference rules stated directly in terms of the combinators defined in problem 1. For each of the inference rules given below state whether or not that inference rule is sound relative to the semantics of higher order logic. Briefly explain each answer.

```
a   (SUBREL ?R ?W)
    ------------------
    (SUBREL ?W ?R)

b   (SUBREL ?R ?R)

c   (SUBREL ?R ?W)
    (SUBREL ?W ?U)
    ------------------
    (SUBREL ?R ?U)

d   (SUBREL ?R (REL-COMP ?R ?R))

e   (SUBREL ?R (REL-UNION ?R ?W))

e   (SUBREL ?R (KLEENE-STAR ?R))

f   ((EVERY (APPLY ?R ?U)) ?U)
    ------------------
    ((EVERY (APPLY (KLEENE-STAR ?R) ?U)) ?U)
```

Problem 5. Consider a dungeon with all exits locked. In other words, any open door connects two rooms inside the dungeon. We can infer that a person who can not get through locked doors can not get out of the dungeon. Formalize this inference in higher order logic in such a way that it can be seen as a special case of of inference rule f given in the previous problem.