CS245 Tutorial

Nov. 16, 2015

1. This question (and the question on the assignment) uses a simple representation of non-negative integers: in binary notation with the bits entered in a basic list. We shall use constants *zero* and *one*, respectively, for the bits 0 and 1. (In Scheme, we use `'zero` and `'one`.) We shall create the lists with the ***least***-significant bit at the start, to make it easier to work with. For example, the list $\langle zero, one \rangle$ denotes the number with binary representation "10"—the number two.

   Consider the following Scheme program, which increments a number in this representation. (Ignore what happens if the input is not a valid number.)

```
( define ( incr a )
        ( cond ( ( equal? a empty ) ( cons 'one empty ) )
               ( ( equal? ( first a ) 'zero ) ( cons 'one ( rest a ) ) )
               ( #t  ( cons 'zero ( incr ( rest a ) ) ) )
        ) )
```

   (a) Write the formulas describing a binary relation $\mathcal{R}_{incr}$ such that $\mathcal{R}_{incr}(a, b)$ holds if and only if `(incr a)` produces value `b`. Show your work: give both the form given directly by the rules, and a simplified form.

   (b) Using your formulas, show that `incr` is total; that is, that $\forall x \cdot \exists y \cdot \mathcal{R}_{incr}(x, y)$. You need not give a fully formal proof, but argue it carefully.

   Remember: the list has the ***least***-significant digit first. Also, it may have "leading zeroes"—for example, the list $\langle zero, one, zero, zero \rangle$ is an alternative representation of the number 2.

(a) *Write the formulas describing a binary relation $\mathcal{R}_{\texttt{incr}}$ such that $\mathcal{R}_{\texttt{incr}}(a, b)$ holds if and only if* `(incr a)` *produces value* `b`*. Show your work: give both the form given directly by the rules, and a simplified form.*

Solution:

Line 1: `( cond ( ( equal? a empty ) ( cons 'one empty ) ) )`

- Direct: $((a = e) \rightarrow \mathcal{R}_{\texttt{incr}}(a, \langle one \rangle))$.
- Simplified: $\mathcal{R}_{\texttt{incr}}(e, \langle one \rangle)$.

Line 2: `( cond ( ( equal? ( first a ) 'zero ) ( cons 'one ( rest a ) ) ) )`

- Direct:

$$((a \neq e \wedge \mathcal{R}_{\texttt{first}}(a, a_f) \wedge a_f = zero \wedge \mathcal{R}_{\texttt{rest}}(a, a_r)) \rightarrow \mathcal{R}_{\texttt{incr}}(a, cons(one, a_r))$$

- Simplified:

$$\mathcal{R}_{\texttt{incr}}(cons\,(zero, a_r)\,, cons\,(one, a_r))$$

Line 3: `( cond ( #t ( cons 'zero ( incr ( rest a ) ) ) ) )`

- Direct:

$$((a \neq e \wedge \mathcal{R}_{\texttt{first}}(a, a_f) \wedge a_f \neq zero \wedge \mathcal{R}_{\texttt{rest}}(a, a_r) \wedge \mathcal{R}_{\texttt{incr}}(a_r, v)) \rightarrow \mathcal{R}_{\texttt{incr}}(a, cons\,(zero, v))$$

- Simplified:

$$(\mathcal{R}_{\texttt{incr}}(x, v) \rightarrow \mathcal{R}_{\texttt{incr}}(cons\,(one, x)\,, cons\,(zero, v)))$$

The above formulas may also be given in "if and only if" form. As given, however, they do suffice for part (b).

(b) *Using your formulas, show that* `incr` *is total; that is, that* $\forall x.\exists y.\mathcal{R}_{\texttt{incr}}(x,y)$. *You need not give a fully formal proof, but argue it carefully.*

Solution: We use induction on the construction of the list, via Axiom BL3, with $\exists y.\mathcal{R}_{\texttt{incr}}(x,y)$ as the formula φ.

The basis is $\exists y.\mathcal{R}_{\texttt{incr}}(e,y)$, which follows from the formula for line 1.

For the inductive step, we assume $\exists y.\mathcal{R}_{\texttt{incr}}(x,y)$ and wish to prove $\forall z.\exists y.\mathcal{R}_{\texttt{incr}}(\mathit{cons}(z,x),y)$. Let u be the value such that $\mathcal{R}_{\texttt{incr}}(x,u)$. In the case that z is *zero*, the formula from line 2 provides the value $\mathit{cons}(\mathit{one},x)$ of y for which $\mathcal{R}_{\texttt{incr}}(\mathit{cons}\,z,x,y)$. Otherwise, the formula from line 3 implies $\mathcal{R}_{\texttt{incr}}(\mathit{cons}(z,x),\mathit{cons}(\mathit{zero},u))$.

2. For each pair of literals, give a unifier if it exists.

a.	$P(a, b, b)$	$\neg P(x, y, z)$
b.	$Q(y, g(a, b))$	$\neg Q(g(x, x), y)$
c.	$R(f(y), y)$	$\neg R(f(x), a)$
d.	$S(f(y), y)$	$\neg S(x, x)$

Unifiers:

a. $[a/x][b/y][b/z]$; i.e., first apply the substitution $[a/x]$, then $[b/y]$, then $[b/z]$. This is the same as $[a/x; b/y; b/z]$.

b. fails. If we make the substitution $[g(x, x)/y]$ we are left with $Q(g(x, x), g(a, b))$, $\neg Q(g(x, x), g(x, x))$, but now x has to be bound to both a and b, which fails.

c. $[y/x][a/y]$; i.e., first apply the substitution $[y/x]$, then $[a/y]$. Alternatively, $[x/y][a/x]$; i.e., first apply the substitution $[x/y]$, then $[a/x]$. Both are the same as $[a/x; a/y]$.

d. fails. If we make the substitution $[f(y)/x]$ we are left with $S(f(y), y)$, $\neg S(f(y), f(y))$ but now y has to be bound to a term that contains y; i.e., $f(y)$. (Notice what would happen if we went ahead and did the substitution $[f(y)/y]$: the two literals would not be identical.)

3. A common error among students is to suppose that, in unification, one is allowed to substitute a term for a constant instead of for a variable. For instance, they will say that the literals $P(c1)$ and $\neg P(a)$ can be unified using the substitution $[c1/a]$, where $c1$ is a Skolem constant and a is a constant. (Recall that substitutions are always of the form $[t/x]$, where t is a term and x is a variable.) Give an example where this leads to an invalid inference.

   - From premise $P(a)$ we can prove the conclusion $\forall x \cdot P(x)$ using resolution refutation, if we use this mistake.
   - Negating the conclusion and converting to CNF gives:

     $$\begin{aligned} &\neg(\forall x \cdot P(x)) \\ &\quad \equiv \quad \exists x \cdot \neg P(x) \\ &\quad \equiv \quad \neg P(c1) \end{aligned}$$

   - Resolution refutation "proof":

1.	$P(a)$	premise
2.	$\neg P(c1)$	negated conclusion
3.	$\bot$	1, 2, $[c1/a]$

4. Consider the following:

   Everyone who loves all animals is loved by someone.
   Anyone who kills an animal is loved by no one.
   Jack loves all animals.
   Either Jack or Curiosity killed the cat, who is named Tuna.
   Did Curiosity kill the cat?

   a. Represent the sentences and the negated conclusion in predicate logic.

      We will use the following constant and predicate symbols:
      Constants: Jack, Curiosity, Tuna
      Predicates: $Animal(x)$, $Loves(x, y)$, $Kills(x, y)$, $Cat(x)$

      1. $\forall x \cdot ((\forall y \cdot Animal(y) \rightarrow Loves(x, y)) \rightarrow (\exists y \cdot Loves(y, x)))$
      2. $\forall x \cdot ((\exists z \cdot (Animal(z) \wedge Kills(x, z))) \rightarrow (\forall y \cdot \neg Loves(y, x)))$
      3. $\forall x \cdot (Animal(x) \rightarrow Loves(Jack, x))$
      4. $Kills(Jack, Tuna) \vee Kills(Curiosity, Tuna)$
      5. $Cat(Tuna)$
      6. $\forall x \cdot (Cat(x) \rightarrow Animal(x))$
      $\neg$7. $\neg Kills(Curiosity, Tuna)$

   b. Convert the formulas to CNF.

      1a. $Animal(f(x)) \vee Loves(g(x), x)$
      1b. $\neg Loves(x, f(x)) \vee Loves(g(x), x)$
      2. $\neg Animal(z) \vee \neg Kills(x, z) \vee \neg Loves(y, x)$
      3. $\neg Animal(x) \vee Loves(Jack, x)$
      4. $Kills(Jack, Tuna) \vee Kills(Curiosity, Tuna)$
      5. $Cat(Tuna)$
      6. $\neg Cat(x) \vee Animal(x)$
      $\neg$7. $\neg Kills(Curiosity, Tuna)$

      Here $f(x)$ and $g(x)$ are Skolem functions.

c. Give a resolution refutation proof that Curiosity killed the cat.

Note that (1a), (1b), ..., (¬7) are just a restatement from the previous page.

1a.	$Animal(f(x)) \vee Loves(g(x), x)$	premise
1b.	$\neg Loves(x, f(x)) \vee Loves(g(x), x)$	premise
2.	$\neg Animal(z) \vee \neg Kills(x, z) \vee \neg Loves(y, x)$	premise
3.	$\neg Animal(x) \vee Loves(Jack, x)$	premise
4.	$Kills(Jack, Tuna) \vee Kills(Curiosity, Tuna)$	premise
5.	$Cat(Tuna)$	premise
6.	$\neg Cat(x) \vee Animal(x)$	premise
¬7.	$\neg Kills(Curiosity, Tuna)$	negated conclusion
8.	$Kills(Jack, Tuna)$	4, ¬7
9.	$\neg Animal(Tuna) \vee \neg Loves(y, Jack)$	2, 8, $[Jack/x; Tuna/z]$
10.	$\neg Cat(Tuna) \vee \neg Loves(y, Jack)$	6, 9, $[Tuna/x]$
11.	$\neg Loves(y, Jack)$	5, 10
12.	$\neg Loves(Jack, f(Jack))$	1b, 11, $[Jack/x; g(Jack)/y]$
13.	$\neg Animal(f(Jack))$	3, 12, $[f(Jack)/x]$
14.	$Loves(g(Jack), Jack)$	1a, 13, $[Jack/x]$
15.	$\perp$	11, 14, $[g(Jack)/y]$