Computer Science 81, Spring 2009
Assignment 7
Due Mon. March 9
Numbers and Computability

1. [40 points] Show that, for any Turing machine M and corresponding input tape X, there is a set of clauses S such that S is unsatisfiable iff M halts on X. Demonstrate using Otter.

   Suggestion: Use a 1-ary function symbol for each distinct tape symbol. The clauses would define the possible moves of the Turing machine. (Note that you will need to represent blank tape and tape expansion.) This is similar to the pegs puzzle example given in class. Clauses for it can be found on the course web page as Pegs Puzzle.

   Demonstrate your technique on a specific M and x by giving the corresponding clauses to Otter and have it check unsatisfiability. Do this for both halting and non-halting cases. The examples to use are given by the two TM tables below. The tape alphabet is {**a**, **c**, **b**} where **b** represents blank. It is assumed the tape has some number of contiguous **a**'s and that the head is positioned at the rightmost **a**, if there is any. Program **double** is supposed to double the number of contiguous **a**'s, which it does by adding a **c** at the right end for each **a**, after replacing that **a** with an **c**. Once each **a** has produced two **c**'s, it converts all **c**'s back to **a**'s. If the head starts over a **b**, that indicates that there are no **a**'s to double. State s is the start state. Program **damaged** is a damaged version of **double**, which doesn't converge. In it, the symbols shown in parentheses in the table replace the other symbols. Show your work with input **baaaa**.

double				
Current state	Symbol read	Symbol written	Head moves	Next state
s	b	b	left	t
s	a	c	right	r
t	b	b	right	q
t	a	c	right	r
t	c	c	left	t
r	c	c	right	r
r	b	c (b)	left	t
q	c	a	right	q
q	b	b	left	e (s)

2. [40 points] Prove using natural deduction the following theorem (associativity of multiplication •) in the theory of the natural numbers:

   $\vdash \forall x\ \forall y\ \forall z\ (x \bullet (y \bullet z) = (x \bullet y) \bullet z)$

   Use the following axioms (called PA for "Peano Arithmetic"), in addition to rules and axioms for equality.

   a. $\forall x\ \neg(S(x) = 0)$
   b. $\forall x\ \forall y\ (S(x) = S(y) \rightarrow x = y)$
   c. $\forall x\ (x + 0 = x)$
   d. $\forall x\ \forall y\ (x + S(y) = S(x + y))$
   e. $\forall x\ (x \bullet 0 = 0)$
   f. $\forall x\ \forall y\ (x \bullet S(y) = x \bullet y + x)$
   g. For any formula φ:
      $(\varphi(0) \wedge \forall x\ (\varphi(x) \rightarrow \varphi(S(x))) \rightarrow \forall x\ \varphi(x)$ [Induction axioms]

On the next pages, we provide some examples of proofs of properties for +.

3. [20 points] Using Otter, prove the base case and the induction step for *associativity* of multiplication. You may introduce as assumptions any derivable formula about +, but you must derive from the axioms given anything about multiplication.

Examples: For a theorem such as commutativity of addition

Lemma 1: $\forall x\ (0 + x = x)$

Proof:

1.	$\forall x\ (x + 0 = x)$	Axiom c
2.	$0 + 0 = 0$	1, $\forall$E
3.	x_0	fresh variable
4.	$0 + x_0 = x_0$	Assumption
5.	$\forall x\ \forall y\ (x + S(y) = S(x + y))$	Axiom d
6.	$\forall y\ (0 + S(y) = S(0 + y))$	4, $\forall$E
7.	$(0 + S(x_0) = S(0 + x_0))$	5, $\forall$E
8.	$(0 + S(x_0) = S(x_0)$	7, 3, Equality
9.	$(0 + x_0 = x_0) \rightarrow (0 + S(x_0) = S(x_0)$	4-8, $\rightarrow$I
10.	$\forall x\ ((0 + x = x) \rightarrow (0 + S(x) = S(x))$	3-9, $\forall$I
11.	$(0 + 0 = 0) \wedge \forall x\ ((0 + x = x) \rightarrow (0 + S(x) = S(x))$	2, 10, $\wedge$I
12.	$\forall x\ (0 + x = x)$	11, Axiom g, $\rightarrow$E

Lemma 2: $\forall y\ \forall x\ (S(x) + y = S(x + y))$

Proof:

1.	$\forall x\ (x + 0 = x)$	Axiom c
2.	x_0	fresh variable
3.	$x_0 + 0 = x_0$	1, $\forall$E
4.	$S(x_0) = S(x_0)$	Equality
5.	$S(x_0+0) = S(x_0)$	3, 4, Equality
6.	$S(x_0)+0 = S(x_0)$	1, $\forall$E
7.	$S(x_0)+0 = S(x_0+0)$	5, 6, Equality
8.	$\forall x\ (S(x) + 0 = S(x+0))$	2-7, $\forall$I

[Lines 1-8 establish the basis for induction. Next we do the induction step.]

9.	y_1	fresh variable
10.	$\forall x\ (S(x) + y_1 = S(x + y_1))$	Assumption
11.	x_1	fresh variable
12.	$S(x_1) + y_1 = S(x_1 + y_1)$	10, $\forall$E
13.	$S(S(x_1) + y_1) = S(S(x_1 + y_1))$	12, Equality
14.	$\forall y\ (S(x_1) + S(y) = S(S(x_1) + y))$	Axiom d, $\forall$E
15.	$S(x_1) + S(y_1) = S(S(x_1) + y_1)$	14, $\forall$E
16.	$\forall y\ (x_1 + S(y) = S(x_1 + y))$	Axiom d, $\forall$E
17.	$x_1 + S(y_1) = S(x_1 + y_1)$	16, $\forall$E
18.	$S(x_1 + S(y_1)) = S(S(x_1 + y_1))$	17, Equality
19.	$S(x_1) + S(y_1) = S(x_1 + S(y_1))$	15, 18, Equality
20.	$\forall x\ (S(x) + S(y_1) = S(x + S(y_1)))$	11-19, $\forall$I
21.	$\forall x\ (S(x) + y_1 = S(x + y_1))$ $\rightarrow \forall x\ (S(x) + S(y_1) = S(x + S(y_1)))$	10-20, $\rightarrow$I
22.	$\forall y\ (\forall x\ (S(x) + y = S(x + y)) \rightarrow \forall x\ (S(x) + S(y) = S(x + S(y))))$	9-21, $\forall$I

[Lines 9-22 constitute the induction step. Next we put this together with the basis, then apply the induction axiom.]

23.	$\forall x\ (S(x) + 0 = S(x+0))$ $\wedge\ \forall y\ (\forall x\ (S(x) + y = S(x + y)) \rightarrow \forall x\ (S(x) + S(y) = S(x + S(y))))$	8, 22, $\wedge$I
24.	$\forall y\ \forall x\ (S(x) + y = S(x + y)))$	23, Axiom g, $\rightarrow$E

[Note that Axiom c is used twice in the above derivation.]

Theorem 1: $\forall y\, \forall x\, (x + y = y + x)$

Proof:

1.	x_0	fresh variable
2.	$x_0 + 0 = x_0$	Axiom c, $\forall$E
3.	$0 + x_0 = x_0$	Lemma 1, $\forall$E
4.	$x_0 + 0 = 0 + x_0$	2, 3, Equality
5.	$\forall x\, (x + 0 = 0 + x)$	1-4, $\forall$I

[The above completes the basis for the induction. Next is the induction step.]

6.	y_1	fresh variable
7.	$\forall x\, (x + y_1 = y_1 + x)$	Assumption
8.	x_1	fresh variable
9.	$x_1 + y_1 = y_1 + x_1$	7, $\forall$E
10.	$S(x_1 + y_1) = S(y_1 + x_1)$	9, Equality
11.	$\forall y\, (x_1 + S(y) = S(x_1 + y))$	Axiom d, $\forall$E
12.	$x_1 + S(y_1) = S(x_1 + y_1)$	11, $\forall$E
13.	$\forall x\, (S(x) + y_1 = S(x + y_1))$	Lemma 2, $\forall$E
14.	$S(x_1) + y_1 = S(x_1 + y_1)$	13, $\forall$E
15.	$x_1 + S(y_1) = S(x_1) + y_1$	10, 14, Equality
16.	$\forall x\, (S(x) + x_1 = S(x + x_1))$	Lemma 2, $\forall$E
17.	$S(y_1) + x_1 = S(y_1 + x_1)$	16, $\forall$E
18.	$S(y_1) + x_1 = S(x_1 + y_1)$	9, Equality
19.	$x_1 + S(y_1) = S(y_1) + x_1$	12, 18, Equality
20.	$\forall x\, (x + S(y_1) = S(y_1) + x)$	8-19, $\forall$I
21.	$\forall x\, (x + y_1 = y_1 + x) \rightarrow \forall x\, (x + S(y_1) = S(y_1) + x)$	7-20, $\rightarrow$I
22.	$\forall y(\forall x\, (x + y = y + x) \rightarrow \forall x\, (x + S(y) = S(y) + x))$	6-21, $\forall$I

[Lines 6-22 constitute the induction step. Next we put this together with the basis, then apply the induction axiom.]

23.	$\forall x\, (x + 0 = 0 + x)$ $\wedge \forall y(\forall x\, (x + y = y + x) \rightarrow \forall x\, (x + S(y) = S(y) + x))$	5, 22, $\wedge$I
24.	$\forall y \forall x\, (x + y = y + x)$	23, Axiom g, $\rightarrow$E

[Note that Lemma 2 is used twice in the above derivation.]

Here is how we might set up the proof of Theorem 1 in Otter. Note that the two lemmas are assumed to be proved separately. Also, the proof likely won't require axioms a and b.

```
% Proving the Commutative Law of Addition using      Induction

set(auto).

formula_list(usable).

all x (s(x) != 0).                                    % Axiom    a

all x all y ((s(x) = s(y) -> x = y)).                 % Axiom    b

all x (x + 0 = x).                                    % Axiom    c

all x all y (x + s(y) = s(x + y)).                    % Axiom    d

all x (0 + x = x).                                    % Lemma    1

all y all x ((s(x) + y = s(x + y))).                  % Lemma    2

% Induction axiom specialized to the commutative law of addition

 (
  (all x (x + 0 = 0 + x))                             % basis
 &
  (all y (
      (all x (x + y = y + x))                         % induction step
   -> (all x (x + s(y) = s(y) + x)))
  )
 )

-> (all y all x (x + y = y + x)).                     % induction axiom


% The following is our theorem negated.

-(all y all x (x + y = y + x)).

end_of_list.
```

Here is the clause set that Otter generates from the above. Note that c1…c4 are Skolem constants, resulting from the existential quantifiers that result from negating the universal quantifiers.

1. s(x)!=0.
2. s(x)!=s(y)|x=y.
3. x+0=x.
4. x+s(y)=s(x+y).
5. 0+x=x.
6. s(x)+y=s(x+y).
7. 0+$c1!=$c1+0|$c3+y=y+$c3|x+x1=x1+x.
8. 0+$c1!=$c1+0|s($c3)+$c2!=$c2+s($c3)|x+x1=x1+x.
9. x+$c4!=$c4+x.

Here is the resolution proof that Otter finds. The numbers on the left are the numbers of clauses as they are generated. (Many generated clauses are not useful in the ultimate proof.) Inside brackets are the clauses that are used in the inference, as well as specific literals indicated by dots. Empty brackets indicate a given clause.

```
2     [] s(x)!=s(y)|x=y.
3     [] 0+$c1!=$c1+0|$c3+x=x+$c3|x+y=y+x.
4     [] 0+$c1!=$c1+0|s($c3)+$c2!=$c2+s($c3)|x+x=x+x.
5     [] x+$c4!=$c4+x.
7,6   [] x+0=x.
8     [] x+s(y)=s(x+y).
10,9  [copy,8,flip.1] s(x+y)=x+s(y).
12,11 [] 0+x=x.
13    [] s(x)+y=s(x+y).
14    [copy,13,demod,10] s(x)+y=x+s(y).
15    [back_demod,4,demod,12,7]
         $c1!=$c1|s($c3)+$c2!=$c2+s($c3)|x+x=x+x.
16    [back_demod,3,demod,12,7] $c1!=$c1|$c3+x=x+$c3|x+y=y+x.
17    [copy,14,flip.1] x+s(y)=s(x)+y.
19    [para_into,6.1.1,6.1.1] x=x.
20    [para_into,6.1.1,2.2.1,demod,7] x=y|s(y)!=s(x).
27    [para_from,20.1.1,5.1.1.1] x+$c4!=$c4+x|s(x)!=s(x).
94    [para_from,15.3.1,5.1.1,unit_del,19,19]
         s($c3)+$c2!=$c2+s($c3).
136   [hyper,16,19] $c3+x=x+$c3|x+y=y+x.
272   [para_into,94.1.1,14.1.1,flip.1] $c2+s($c3)!=$c3+s($c2).
313   [para_into,272.1.1,17.1.1] s($c2)+$c3!=$c3+s($c2).
3994  [hyper,136,27,19] $c3+x=x+$c3.
3995  [copy,3994,flip.1] x+$c3=$c3+x.
3996  [binary,3995.1,313.1] $F.
```

Demod and *para* stand for demodulation and paramodulation, which are optimized rewriting rules that Otter uses to handle equality. *Hyper* refers to "hyper-resolution", which resolves multiple clauses in one step. A brief summary of these rules is given in:

http://www.cs.unm.edu/~mccune/mace4/manual/Dec-2007/glossary.html