Computer Science 81, Spring 2010
Assignment 8
Due Tue. March 30
Computation as Logic

Show by example 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 (e.g. the web-accessible Otter-λ: http://www.michaelbeeson.com/research/otter-lambda/)

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)

A manual for Otter can be found here:
http://www-unix.mcs.anl.gov/AR/otter/otter33.pdf

Other Otter-related info and examples are on the course website. You should try some of them before constructing your solution.

Note on Variables in Otther: Determining whether a simple term is a constant or a variable depends on the context of the term. If it occurs in a clause, the symbol determines the type: the default rule is that a simple term is a variable if it starts with `u`, `v`, `w`, `x`, `y`, or `z`. If the flag `prolog_style_variables` is set, a simple term is a variable if and only if it starts with an upper-case letter or with `_`. (Therefore, variables in clauses must be ordinary names.) In a formula, a simple term is a variable if and only if it is bound by a quantifier.