COMP102P. Model checking coursework

Robin Hirsch

November 26, 2015

This is part of a compulsory coursework on model checking. You will be asked to upload the code of your parser and model checker by 15th January 2016. Instructions for uploading your code will come later.

In this part of the coursework you have to parse first order formulas in a language with no function symbols and one binary predicate symbol 'X' denoting the edge relation in a graph, so $X[xy]$ means that there is an edge from x to y. A binary connective is a character 'v', '^' or '>' (denoting or, and, implies). A variable is a character 'x', 'y' or 'z' (three variables should be enough for this coursework). There are no constants and no function symbols, so a term is just a variable. A formula is defined by

$$\phi ::= X[ts] \mid -\phi \mid (\phi \circ \phi) \mid Ez\ \phi \mid Az\ \phi$$

where $\circ$ is a binary connective, t, s, z are variables. We will not include any spaces in our formulas.

Have a look at the program called "parser.c". Apart from the main method, there are four other methods: parse, partone, parttwo and bin. These have not been implemented, that is your job. Without altering the main method you should implement these other methods. At a unix prompt you should type

gcc -Wall parser.c -o parser.out

and then run your program by typing parser.out. You can test your program by trying various strings and checking that it classifies the strings correctly.

Here are some examples with the correct answer

Formula	Output	part 1	connective	part 2
X[yz]	An atomic formula			
(AxX[xz]>-EzX[zz])	a binary connective formula	AxX[xz]	>	-EzX[zz]
Ez(X[xz]>–(X[zx]^X[yy]))	an existential formula			
X[x]	Not a formula			
((ExX[xx]vAxEyX[xy])^(X[yy]vX[zy]))	a binary connective formula	(ExX[xx]vAxEyX[xy])	^	(X[yy]vX[zy])