CS 598mp: Homework; due on Thu/Fri, 17/18 March

Let us consider LTL with past operators over finite strings.

Fix a finite set of propositions $\mathcal{P}$. Models will be finite (non-empty) words over $2^\mathcal{P}$, i.e. the words in $(2^\mathcal{P})^+$.

LTL is defined as follows:

$$\varphi, \psi ::= p \mid \neg \varphi \mid \varphi \lor \psi \mid X \varphi \mid \varphi S \psi$$

where $p \in \mathcal{P}$.

Intuitively, $X$ stands for “last-state” (as opposed to “next-state” in LTL) and $\varphi S \psi$ stands for “$\varphi$ has been holding since $\psi$ held”.

The formal semantics is defined as follows. Let $w \in (2^\mathcal{P})^+$ be a model and let $|w| = n$. Then, for each $0 \leq i \leq n$, let $w[i]$ denote the $i^{th}$ element in $w$. We define when $(w, i)$ satisfies $\varphi$, for each $i$, by induction over the structure of $\varphi$:

- $(w, i) \models p$ iff $p \in w[i]$
- $(w, i) \models \neg \varphi$ iff $(w, i) \not\models \varphi$
- $(w, i) \models \varphi \lor \psi$ iff $(w, i) \models \varphi$ or $(w, i) \models \psi$
- $(w, i) \models X \varphi$ iff $i > 0$ and $(w, i - 1) \models \varphi$
- $(w, i) \models \varphi S \psi$ iff there exists $j \leq i$ such that $(w, j) \models \psi$ and for every $j < i'$, $(w, i') \models \varphi$.

Finally, $w = \varphi$ iff $(w, n - 1) \models \varphi$, i.e. the last position must satisfy $\varphi$. Let $\text{Models}(\varphi) = \{ w \in (2^\mathcal{P})^+ \mid w \models \varphi \}$.

1. Given any formula $\varphi$ in LTL, show how to construct an automaton $A_\varphi$ on finite words over $2^\mathcal{P}$ that accepts exactly the models of $\varphi$.
   (Take care to define it using the same approach used in class; i.e. building it using atoms and with local conditions that successive states should satisfy.)

2. Show that the above automaton is deterministic.
   (If it is not, construct one that is!)

[The above automaton built can be used as a monitor to detect errors in a system. Suppose we are given a property $\varphi$ which we want all runs of the system to satisfy. Then we can let the system run and observe the states and simulate the above automaton for $\varphi$ on the run. If at any point we reach a non-accepting state, the property $\varphi$ does not hold, and we can detect the error immediately. The memory used by this algorithm will only be linear in the formula.]