Model Checking LTL by Buchi Automata

1 Setup and high-level solution

We are given a Transition System τ and an LTL φ, both over the same set of atomic propositions AP. The task is to decide if $\tau \models \varphi$

That is, if all runs of τ satisfy φ, i.e. $Traces(\tau) \subseteq Words(\varphi)$.
Equivalently

$$Traces(\tau) \cap \left((2^{AP})^{\omega} \backslash Words(\varphi)\right) = \varnothing$$

This is the same as

$$Traces(\tau) \cap Words(\neg\phi) = \varnothing$$

So if both the runs of τ and the models of φ are represented by Buchi Automata, we can construct the intersection and check for emptiness. If it is empty, reply that $\tau \models \varphi$, otherwise return $a, b \in (2^{AP})^*$ such that ab^{ω} is a run of τ that falsifies φ

2 Difficulties of operations on BA

1. **Transforming a TS into a Buchi Automaton** - easy - move each label from the state onto all outgoing transitions and introduce a new start state (of the automaton) instead of multiple ones (of the TS)
2. **Constructing the intersection of two BA** - easy - take the product of the two BA and making sure that we alternate infinitely often between accepting states from the two
3. **Checking a BA for emptiness** - easy - check if there is a non-trivial strongly connected component that contains an accepting state and is reachable from the start state
4. **Transforming an LTL formulae into an equivalent BA** - difficult - Translate it into a generalised BA that has multiple sets of accepting states and the acceptance condition is that a state form each set is visited infinitely often. A GBA is however easily transformed into an equivalent BA
5. **Negating a BA** - Difficult

3 LTL to Buchi

3.1 States

A state has two components

1. A subset of the AP (that are true; all the other are false) that records the current world, which is the last input seen
2. A subset of all subformulae of the φ that should be true in the future, from this state onward

- A state must be propositionally consistent. This takes care of all boolean connectives

3.2 Transitions

A transition from s_1 into s_2 labelled by $a \in 2^{AP}$ is added if

1. The label a matches the first component of the state s_2
2. The state s_1 has the subformula $\circ\psi$ iff the state s_2 has the subformula (or AP) ψ - this takes care of the Next operator
3. Whether the state s_2 has the subformula $\varphi_1 \cup \varphi_2$
   (a) The state s_1 has φ_2, or
   (b) The state s_1 has φ_1 and the state s_2 has $\varphi_1 \cup \varphi_2$

   This partly takes care of the until operator as it can be expanded as follows

$$\varphi_1 \cup \varphi_2 \rightarrow \varphi_2 \vee (\varphi_1 \wedge \bigcirc(\varphi_1 \cup \varphi_2))$$

3.3 Acceptance and Start

The expansion $\varphi_1 \cup \varphi_2 \to \varphi_2 \vee (\varphi_1 \wedge \bigcirc(\varphi_1 \cup \varphi_2))$ doesn't guarantee that φ_2 eventually happens. Thus, any infinite run that always has $\varphi_1 \cup \varphi_2$ but never φ_2 is inconsistent. We can prevent it, though, by insisting that every run has states that have φ_2 or haven't $\varphi_1 \cup \varphi_2$ infinitely often.