8 APPENDIX

Section 8.1 describes how an STL formula can be converted into an equivalent RFOL formula. Section 8.2 contains the proofs for the Theorems of Section 4

8.1 Transforming STL formulae in RFOL

We show that any STL formula can be expressed in RFOL. Let x be a real variable and $c \in \mathbb{R}$, an STL formula can be described using the following grammar:

$$\phi \quad ::= \quad x \sim c \mid \neg\phi \mid \phi_1 \vee \phi_2 \mid \phi_1 \wedge \phi_2 \mid \mathcal{F}_{[a,b]}\,\phi \mid \mathcal{G}_{[a,b]}\,\phi \mid \phi_1\,\mathcal{U}_{[a,b]}\,\phi_2 \mid \phi_1\,\mathcal{R}_{[a,b]}\,\phi_2$$

where $x \in X$, $\sim \in \{<, \leq\}$. $\neg, \wedge \vee$ are boolean operators while $\mathcal{U}_{[a,b]}$, $\mathcal{F}_{[a,b]}$ are temporal operators.

Any STL can be expressed in RFOL by executing a procedure that follows two steps.

`Step 1.` The STL formula is rewritten into an equivalent STL formula by executing the following operations:

- compute the negation normal form (NNF) of the STL formula;[2]
- transform any atom in the form $\neg(x < c)$ into $x \geq c$ and $\neg(x \leq c)$ into $x > c$.

Note that the previous procedure pushes all the negations into formula atoms.

`Step 2.` The syntax tree of the STL formula is analyzed from the root to the leaves and every operator of the resulting STL formula is transformed into RFOL as follows:

- every STL subformula φ in the form $x \sim c$, $\phi_1 \vee \phi_2$ and $\phi_1 \wedge \phi_2$ is converted into an equivalent RFOL subformula that uses the same boolean or relational operator;
- every STL subformula φ in the form $\mathcal{F}_{[a,b]}\,\phi$ is converted into
  (1) the RFOL formula $\exists t \in [a,b] : \phi$ if φ is *not* nested into another formula that uses any temporal operators;
  (2) the RFOL formula $\exists t \in [t_f + a, t_f + b] : \phi$ if φ is nested into another formula that uses temporal operators. The timed variable t_f is the timed variable introduced in the RFOL formula when the other formula is mapped into RFOL;
- every STL subformula φ in the form $\phi_1\,\mathcal{U}_{[a,b]}\,\phi_2$ is converted into
  (1) the RFOL formula $\exists t \in [a,b] : (\phi_2 \wedge (\forall t' \in [a,t] : \phi_1))$ if φ is *not* nested into another formula that uses the temporal operators;
  (2) the RFOL formula $\exists t \in [t_f + a, t_f + b] : (\phi_2 \wedge (\forall t' \in [t_f + a, t] : \phi_1))$ if φ is nested into another formula that uses temporal operators. The timed variable t_f is the timed variable introduced in the RFOL formula when the other formula is mapped into RFOL;
- every STL subformula φ in the form $\mathcal{G}_{[a,b]}\,\phi$ is converted into
  (1) the RFOL formula $\forall t \in [a,b] : \phi$ if φ is *not* nested into another formula that uses temporal operators;
  (2) the RFOL formula $\forall t \in [t_f + a, t_f + b] : \phi$ if φ is nested into another formula that uses temporal operators. The timed variable t_f is the timed variable introduced in the RFOL formula when the other formula is mapped into RFOL;
- every STL subformula φ in the form $\phi_1\,\mathcal{R}_{[a,b]}\,\phi_2$ is converted into
  (1) the RFOL formula $\exists t \in [a,b] : ((\phi_2 \wedge \phi_1) \wedge (\forall t' \in [a,t] : \phi_2)) \vee \forall t \in [a,b] : (\phi_2)$ if the formula φ is *not* nested into another formula that uses the temporal operators;
  (2) the RFOL formula $\exists t \in [t_f + a, t_f + b] : ((\phi_2 \wedge \phi_1) \wedge (\forall t' \in [t_f + a, t] : \phi_2)) \vee \forall t \in [t_f + a, t_f + b] : \phi_2$ if φ is nested into another formula that uses temporal operators. The timed variable t_f is the timed variable introduced in the RFOL formula when the other formula is mapped into RFOL.

Note that it is trivial to show correctness of the proposed encoding as the generated RFOL formulae are a direct encoding of the semantics of the corresponding STL formulae.

8.2 Proofs

We provide a sketch of the proofs for the theorems presented in Section 4.

Theorem 8.1. *Let φ be an RFOL formula and let $\varphi_{\Uparrow}$ be its* shifted-formula. *For any signal set F, we have: $[\![\varphi]\!]_F = [\![\varphi_{\Uparrow}]\!]_F$*

Proof Sketch. The proof follows from the fact that every time interval and every time variable are shifted consistently together, i.e., if the bound of a variable t is changed from $\langle n_1, n_2\rangle$ into $\langle n_1 + d, n_2 + d\rangle$ the occurrences of t in the formula are replaced with $t - d$ ensuring that the correct value of signal is considered in the evaluation of the formula. Hence, the semantics of formulas is not impacted. □

Theorem 8.2. *Let M_p be a (partial) Simulink model, and let I be a test input for M_p defined over the time domain $\mathbb{T} = [0, t_u]$. Let φ be a requirement of M_p in RFOL. Suppose $\{O_1, O_2 \ldots O_k\} = H_p(I, M_p)$ and $\{\{e_1\}, \{e_2\}, \ldots, \{e_k\}\} = H_p(I, M_p + M_\varphi)$ are simulation results generated for the time domain $\mathbb{T}$. Then, the value of φ over every signal set $O_i \in \{O_1, O_2 \ldots O_k\}$ is equal to the value of the signal e_i generated by $M_p + M_\varphi$ at time t_u. That is, $[\![\varphi]\!]_{O_i} = e_i(t_u)$. Further, we have:*

$$oracle(M_p, I, \varphi) = \min_{e \in \{e_1, \ldots, e_k\}} e(t_u)$$

That is, the minimum value of the outputs of $M_p + M_\varphi$ at t_u is equal to the oracle value as defined by Definition 3.3.

Proof Sketch. The proof follows by structural induction on RFOL formulas. The atomic terms are constructed by rules 1, 2 and 4. It is easy to show that these rules correctly compute the semantics of these atomic terms, and further, they do not rely on a signal value that is not yet generated up to the current time t. The inductive proof for rules 3, 5, 6, 7, 8, and 9 is as follows: (1) Each rule correctly computes the value of RFOL constructs based on the RFOL semantics, and (2) the rules do not rely on a signal value that is not yet generated up to the current time t. In particular, for Rule8, due to the interval-shifting procedure, we know that the value of ϕ in $\forall t \in \langle \tau_1, \tau_2\rangle : \phi$ at time t is correctly computed inductively.

[2] A formula is in negation formal form if negation $\neg$ occurs only directly in front of atoms.

Finally, by Theorem 8.1, we know that our time and interval shifting procedures are semantic preserving. Therefore, our translation for $\varphi_{\Uparrow}$, i.e., M_φ, is able to correctly compute the semantics of φ. □

Theorem 8.3. *Let M_p be a (partial) Simulink model, and let I be a test input for M_p over the time domain $\mathbb{T} = [0, t_u]$. Let φ be a requirement of M_p in RFOL. Suppose $\{\{e_1\}, \{e_2\}, \ldots, \{e_k\}\} = H_p(I, M_p + M_\varphi)$ are simulation results generated for $\mathbb{T}$. Let d be the maximum constant appearing in the upper bounds of the time intervals of $\varphi_{\Uparrow}$ for existential quantifiers (i.e., time intervals in the form of $\exists t \in [\tau_1, \tau_2] : \phi$ in $\varphi_{\Uparrow}$). Each $e_i \in \{\{e_1\}, \{e_2\}, \ldots, \{e_k\}\}$ is decreasing over the time interval $(d, t_u]$.*

Proof Sketch. By Definition 3.1 φ is closed. Thus, the output of the oracle is the output of a block that is generated using either Rule6, Rule7 or Rule8. Indeed, as there is no free variable, i.e., any variable is scoped by existential and universal quantifiers, the output signal of a block generated using Rule9 cannot be the output of the oracle as it has to be then processed by at least a block generated by Rule8. The output signal generated from the Simulink blocks obtained from the univeral quantifiers by applying Rule 8 is decreasing as at each simulation step they compute the minimum of a previously computed value and a new value which is less than or equal to the previously computed value. Furthermore, if $t > d$, the output signal generated from the Simulink blocks obtained from the existential quantifiers by applying Rule 8 is constant by construction. Finally, the maximum and minimum of two decreasing signals is also decreasing. Thus, each $e_i \in \{\{e_1\}, \{e_2\}, \ldots, \{e_k\}\}$ is decreasing over the time interval $(d, t_u]$. □