Translation rules for code generation from TLA$^+$ specifications

Contents

1 TLA$^+$ syntax

Identifiers I, C Values v Parameters p

$$
\begin{array}{lrcl}
\text{Specification} & \mathit{Spec} & ::= & \mathit{Module}\ M \\
 & & & \text{CONSTANTS}\ C_o,\ \ldots, C_n \\
 & & & \text{VARIABLES}\ V_o,\ \ldots, V_n \\
 & & & D_0,\ \ldots, D_n \\
\text{Definition} & D & ::= & \mathit{Action}(p_0, \ldots, p_n) \triangleq \mathcal{A} \\
\text{Action} & \mathcal{A} & ::= & A \mid P \mid \mathcal{A} \land \mathcal{A} \mid \mathcal{A} \lor \mathcal{A} \\
\text{Condition} & P, Q & ::= & \neg P \mid P \land P \mid P \lor P \mid v_1 \in v_2 \\
 & & \mid & v_1 = v_2 \mid v_1 \neq v_2 \mid \text{ENABLED}\ \mathcal{A} \\
\text{Transition} & T & ::= & I' = v \mid \text{UNCHANGED}\ \langle I_0, \ldots, I_n \rangle \\
\text{Set} & S & ::= & v \mid S_a \cup S_b \\
\text{Record} & R & ::= & [k \mapsto v] \mid [I\ \text{EXCEPT}\ ![k] = v]
\end{array}
$$

2 Elixir syntax

Atoms $\texttt{i}, \texttt{k}$ Values $\texttt{x}, \texttt{y}$ Parmetros $\texttt{p}$

$$\begin{array}{llcl}
\text{State} & \texttt{t} & ::= & \texttt{action(variables, }\overline{\texttt{p}}\texttt{)} \\
 & & | & \texttt{variables} \mid \texttt{Map.merge(a, a)} \\
 & & | & \texttt{\%\{ i}_o : \texttt{x}_o\texttt{, ..., i}_n : \texttt{x}_n \texttt{ \}} \\
\text{Condition} & \texttt{c} & ::= & \texttt{condition(variables, }\overline{\texttt{p}}\texttt{)} \\
 & & | & \texttt{not c} \mid \texttt{c and c} \mid \texttt{c or c} \\
\text{Definition} & \texttt{d} & ::= & \texttt{def action(variables, }\overline{\texttt{p}}\texttt{) do} \\
 & & & \quad \texttt{...} \\
 & & & \texttt{end} \\
 & & & | \\
\texttt{def decide([info])} & & & \\
\text{Set} & \texttt{s} & ::= & \texttt{MapSet.new([x])} \\
 & & | & \texttt{MapSet.union(s}_a\texttt{, s}_b\texttt{)} \\
\text{Record} & \texttt{r} & ::= & \texttt{\%\{ k: x \}} \mid \texttt{Map.put(i, k, x)} \\
\text{Information} & \texttt{i} & ::= & \texttt{\%\{ action: ``Name'', condition: c, state: a \}} \\
 & & | & \texttt{Enum.map(x, f (i) -> [info] end)}
\end{array}$$

3 Top-level Spec translation

$$\boxed{\vdash \mathit{Spec} \rightarrowtail \mathit{code}}$$

$$\frac{\begin{array}{rll}
\Gamma \vdash_{const} C_0, \ldots, C_n \rightarrowtail \texttt{const}_0\texttt{, ..., const}_n & & \\
\{C_0 : const, \ldots, C_n : const\} \vdash_{dec} & Def_0 & \rightarrowtail \texttt{def}_0 \\
 & \vdots & \\
\{C_0 : const, \ldots, C_n : const\} \vdash_{dec} & Def_n & \rightarrowtail \texttt{def}_n \\
\{C_0 : const, \ldots, C_n : const\} \vdash_{next} & Def_{next} & \rightarrowtail \texttt{def\_next} \\
\{M : module, C_0 : const, \ldots, C_n : const\} \vdash_{init} & Def_{init} & \rightarrowtail \texttt{state}
\end{array}}{\vdash \begin{array}{l}
\textsc{module}\ \ M \\
\textsc{constants}\ \ C_0, \ldots, C_n \\
\textsc{variables}\ V_0, \ldots, V_n \\
Def_0 \\
\vdots \\
Def_n \\
Def_{init} \\
Def_{next}
\end{array} \rightarrowtail \begin{array}{l}
\text{defmodule M do} \\
\quad \text{@oracle spawn(Oracle, :listen, [])} \\
\quad \texttt{const}_0\text{, ..., }\texttt{const}_n \\
\quad \texttt{def}_0\text{, ..., }\texttt{def}_n \\
\quad \text{def\_decide} \\
\quad \text{def\_main} \\
\text{end} \\
 \\
\text{M.main(state)}
\end{array}} \ \texttt{(MOD)}$$

4 Definition translation

$$\boxed{\Gamma \vdash_{def} D \rightarrowtail code}$$

$$\frac{\Gamma \cup \{p_0 : param, \ldots, p_n : param\} \vdash_a \mathcal{A} \rightarrowtail (\{c_o, \ldots, c_n\}, \{a_0, \ldots, a_n\})}{\Gamma \vdash_{def} Action(p_0, \ldots, p_n) \triangleq \mathcal{A} \rightarrowtail \begin{array}{l} \text{def action\_condition(variables, } p_0,\ \ldots,\ p_n\text{) do} \\ \quad \text{Enum.all?([}\mathtt{c}_0\text{, ..., }\mathtt{c}_n\text{])} \\ \text{end} \\ \text{def action(variables, } p_0,\ \ldots,\ p_n\text{) do} \\ \quad \text{Map.merge(}a_0\text{, Map.merge(..., }a_n\text{))} \\ \text{end} \end{array}} \; \texttt{(DEF)}$$

5 Action translation

$$\boxed{\Gamma \vdash_a \mathcal{A} \rightarrowtail (\bar{c}, \bar{a})}$$

$$\frac{\Gamma \vdash_p P \rightarrowtail c}{\Gamma \vdash_a P \rightarrowtail (\{c\},\ \{\})}\ \texttt{(COND)} \quad \frac{\Gamma \vdash_t A \rightarrowtail a}{\Gamma \vdash_a A \rightarrowtail (\{\},\ \{a\})}\ \texttt{(TRA)}$$

$$\frac{\begin{array}{c}\Gamma \vdash_a \mathcal{A}_0 \rightarrowtail (\overline{c_0},\ \overline{a_0}) \\ \vdots \\ \Gamma \vdash_a \mathcal{A}_n \rightarrowtail (\overline{c_n},\ \overline{a_n})\end{array}}{\Gamma \vdash_a \begin{array}{c}\wedge\ \mathcal{A}_0 \\ \vdots \\ \wedge\ \mathcal{A}_n\end{array} \rightarrowtail (\overline{c_o} \cup \cdots \cup \overline{c_n},\ \overline{a_o} \cup \cdots \cup \overline{a_n})}\ \texttt{(AND)} \quad \frac{\begin{array}{cc}\Gamma \vdash_a \mathcal{A}_0 \rightarrowtail (\overline{c_0},\ \overline{a_0}) & \Gamma \vdash_i \mathcal{A}_0 \rightarrowtail \overline{\mathtt{i}_0} \\ \vdots & \vdots \\ \Gamma \vdash_a \mathcal{A}_n \rightarrowtail (\overline{c_n},\ \overline{a_n}) & \Gamma \vdash_i \mathcal{A}_n \rightarrowtail \overline{\mathtt{i}_n}\end{array}}{\Gamma \vdash_a \begin{array}{c}\vee\ \mathcal{A}_0 \\ \vdots \\ \vee\ \mathcal{A}_n\end{array} \rightarrowtail \left(\begin{array}{c}\overline{c_o} \cup \cdots \cup \overline{c_n}, \\ \{\ \mathtt{decide(}\overline{\mathtt{i}_0} \cup \cdots \cup \overline{\mathtt{i}_n}\mathtt{)}\ \}\end{array}\right)}\ \texttt{(OR)}$$

$$\frac{\begin{array}{c}\Gamma \vdash_v v_0 \rightarrowtail \mathtt{x}_0 \\ \vdots \\ \Gamma \vdash_v v_n \rightarrowtail \mathtt{x}_n\end{array}}{\Gamma \vdash_a \mathit{Action}(v_0, \ldots, v_n) \rightarrowtail (\{\ \text{action\_condition(variables, } \mathtt{x}_0\text{, ..., } \mathtt{x}_n\text{)}\ \}, \{\ \text{action(variables, } \mathtt{x}_0\text{, ..., } \mathtt{x}_n\text{)}\ \})}\ \texttt{(CALL)}$$

$$\frac{\begin{array}{c}\Gamma \vdash_p P \rightarrowtail c \\ \Gamma \vdash_a \mathcal{A}_t \rightarrowtail (\{\mathtt{ct}_0, \ldots, \mathtt{ct}_n\},\ \{\mathtt{at}_0, \ldots, \mathtt{at}_n\}) \\ \vdots \\ \Gamma \vdash_a \mathcal{A}_t \rightarrowtail (\{\mathtt{ce}_0, \ldots, \mathtt{ce}_n\},\ \{\mathtt{ae}_0, \ldots, \mathtt{ae}_n\})\end{array}}{\Gamma \vdash_a \begin{array}{l}\textsc{if}\ P \\ \textsc{then}\ \mathcal{A}_t \\ \textsc{else}\ \mathcal{A}_e\end{array} \rightarrowtail (\{\mathit{condition}\},\ \{\mathit{transition}\})}\ \texttt{(IF)}$$

where

condition =

```
if c do
  $\mathtt{ct}_0$ and ... and $\mathtt{ct}_n$
else
  $\mathtt{ce}_0$ and ... and $\mathtt{ce}_n$
end
```

transition =

```
if c do
  Map.merge($\mathtt{at}_0$, Map.merge(..., $\mathtt{at}_n$))
else
  Map.merge($\mathtt{ae}_0$, Map.merge(..., $\mathtt{ae}_n$))
end
```

6 Predicate translation

$$\boxed{\Gamma \vdash_p P \rightarrowtail \texttt{c}}$$

$$\frac{\Gamma \vdash_v v_x \rightarrowtail x \quad \Gamma \vdash_v v_y \rightarrowtail y}{\Gamma \vdash_p v_x = v_y \rightarrowtail \texttt{x == y}}\ \texttt{(PRED-EQ)} \quad \frac{\Gamma \vdash_v v_x \rightarrowtail x \quad \Gamma \vdash_v v_y \rightarrowtail y}{\Gamma \vdash_p v_x v_y \rightarrowtail \texttt{x != y}}\ \texttt{(PRED-NEQ)}$$

$$\frac{\Gamma \vdash_v v_e \rightarrowtail e \quad \Gamma \vdash_v v_l \rightarrowtail l}{\Gamma \vdash_p v_e \in v_l \rightarrowtail \texttt{Enum.member?(}l\texttt{, }e\texttt{)}}\ \texttt{(PRED-IN)}$$

$$\frac{\Gamma \vdash_p P \rightarrowtail c}{\Gamma \vdash_p \neg P \rightarrowtail \texttt{not c}}\ \texttt{(PRED-NOT)}$$

$$\frac{\begin{array}{c}\Gamma \vdash_p P_0 \rightarrowtail \texttt{c}_0 \\ \vdots \\ \Gamma \vdash_p P_n \rightarrowtail \texttt{c}_n\end{array}}{\Gamma \vdash_a \begin{array}{l}\vee\ P_0 \\ \quad\vdots \\ \vee\ P_n\end{array} \rightarrowtail \texttt{Enum.any?([c}_0\texttt{, ..., c}_n\texttt{])}}\ \texttt{(PRED-OR)}$$

$$\frac{\Gamma \vdash_a \mathcal{A} \rightarrowtail (\texttt{c}_0, ..., \texttt{c}_n,\ \overline{\texttt{t}})}{\Gamma \vdash_p \textsc{enabled}\ \mathcal{A} \rightarrowtail \texttt{c}_0\ \texttt{and ... and c}_n}\ \texttt{(PRED-EN)}$$

7 Transition translation

$$\boxed{\Gamma \vdash_t T \rightarrowtail \texttt{t}}$$

$$\frac{}{\begin{array}{l}\Gamma \vdash_t \textsc{unchanged}\ \langle I_0, \ldots, I_n\rangle \rightarrowtail \\ \quad \%\{I_0 : \quad \texttt{variables[:}I_0\texttt{]} \\ \qquad\quad \vdots \\ \qquad I_n : \quad \texttt{variables[:}I_n\texttt{]}\}\end{array}}\ \texttt{(TRA-UNCH)}$$

$$\frac{\Gamma \vdash_v v \rightarrowtail x}{\Gamma \vdash_t I' = v \rightarrowtail \texttt{\%\{ }I\texttt{: }x\texttt{ \}}}\ \texttt{(TRA-PRIM)}$$

8 Values translation

$$\boxed{\Gamma \vdash_v v \rightarrowtail x}$$

$$\frac{\{I : \mathit{param}\} \in \Gamma}{\Gamma \vdash_v I \rightarrowtail I}\ \texttt{(VAL-PARAM)} \qquad \frac{\{I : \mathit{param}\} \notin \Gamma \qquad \{I : \mathit{const}\} \notin \Gamma}{\Gamma \vdash_v I \rightarrowtail \texttt{variables[:I]}}\ \texttt{(VAL-VAR)}$$

$$\frac{\begin{array}{c}\{I : \mathit{const}\} \in \Gamma \\ \{M : \mathit{module}\} \notin \Gamma\end{array}}{\Gamma \vdash_v I \rightarrowtail \texttt{@I}}\ \texttt{(VAL-CONST)} \quad \frac{\begin{array}{c}\{I : \mathit{const}\} \in \Gamma \\ \{M : \mathit{module}\} \in \Gamma\end{array}}{\Gamma \vdash_v I \rightarrowtail \texttt{M.I}}\ \texttt{(VAL-ATTR)}$$

$$\frac{\Gamma \vdash_v v_0 \rightarrowtail x_o\ \ \ldots\ \ \Gamma \vdash_v v_n \rightarrowtail x_n}{\Gamma \vdash_v \{v_0, \ldots, v_n\} \rightarrowtail \texttt{MapSet.new([}\texttt{x}_0, \ldots, \texttt{x}_n\texttt{])}}\ \texttt{(SET-LIT)}$$

$$\frac{\Gamma \vdash_v S_a \rightarrowtail s_a \quad \Gamma \vdash_v S_b \rightarrowtail s_b}{\Gamma \vdash_v S_a \cup S_b \rightarrowtail \texttt{MapSet.union(s}_a\texttt{, s}_b\texttt{)}}\ \texttt{(SET-UNION)}$$

$$\frac{\Gamma \vdash_v v_0 \rightarrowtail x_o\ \ \ldots\ \ \Gamma \vdash_v v_n \rightarrowtail x_n}{\Gamma \vdash_v [k_0 \mapsto v_o, \ldots, k_n \mapsto v_n] \rightarrowtail \texttt{\%\{ }k_0\texttt{: x}_0, \ldots, \texttt{k}_n\texttt{: x}_n\texttt{ \}}}\ \texttt{(REC-LIT)}$$

$$\frac{\Gamma \vdash_v v_v \rightarrowtail \texttt{x} \quad \Gamma \vdash_v v_i \rightarrowtail \texttt{i}}{\Gamma \vdash_v [v_i\ \text{EXCEPT}\ ![k] = v_v] \rightarrowtail \texttt{Map.put(i, k, x)}}\ \texttt{(REC-EXCEPT)}$$

$$\frac{\Gamma \vdash_v v \rightarrowtail x}{\Gamma \vdash_v v[k] \rightarrowtail \texttt{x[k]}}\ \texttt{(REC-INDEX)}$$

$$\frac{\Gamma \vdash_v [\overline{k- > v}] \rightarrowtail \texttt{x}_l \quad \Gamma \vdash_v v_r \rightarrowtail \texttt{x}_r \quad \Gamma \vdash_v v_v \rightarrowtail \texttt{x}_v}{\begin{array}{c}\Gamma \vdash_v [I \in v_r- > v_v,\ \overline{k- > v}] \rightarrowtail \\ \texttt{x}_r\ \texttt{|> Enum.map(fn i -> \{i, x}_v\texttt{\} end) |> Enum.into(x}_l\texttt{)}\end{array}}\ \texttt{(REC-EX)}$$

9 Initial state translation

$$\boxed{\Gamma \vdash_{\mathit{init}} P \rightarrowtail \texttt{t}}$$

$$\frac{\Gamma \vdash_v v \rightarrowtail \texttt{x}}{\Gamma \vdash_{\mathit{init}} I = v \rightarrowtail \texttt{\%\{ i: x \}}}\ \texttt{(INIT-EQ)}$$

$$\frac{\begin{array}{c}\Gamma \vdash_{\mathit{init}} P_0 \rightarrowtail \texttt{t}_0 \\ \vdots \\ \Gamma \vdash_{\mathit{init}} P_n \rightarrowtail \texttt{t}_n\end{array}}{\Gamma \vdash_{\mathit{init}} \begin{array}{c}\land\ P_0 \\ \vdots \\ \land\ P_n\end{array} \rightarrowtail \begin{array}{l}\text{Map.merge(}\texttt{t}_0, \\ \quad \text{Map.merge(..., }\texttt{t}_n\text{))}\end{array}}\ \texttt{(INIT-AND)}$$

10 Next state action translation

$$\boxed{\Gamma \vdash_{next} \mathit{Action} \triangleq \mathcal{A} \rightarrowtail \mathit{code}}$$

$$\frac{\Gamma \vdash_a \mathcal{A} \rightarrowtail (\{\},\ \texttt{t}_0, \ldots, \texttt{t}_n)}{\begin{array}{l}\Gamma \vdash_{next} \mathit{Action} \triangleq \mathcal{A} \rightarrowtail \\ \quad \text{def main(variables) do} \\ \qquad \text{IO.puts (inspect variables)} \\ \\ \qquad \text{main(} \\ \qquad\quad \text{Map.merge(}\texttt{t}_0\text{, Map.merge(..., }\texttt{t}_n\text{))} \\ \qquad \text{)} \\ \quad \text{end}\end{array}} \;\texttt{(NEXT)}$$

11 Information extraction

$$\boxed{\Gamma \vdash_i \mathcal{A} \rightarrowtail \texttt{info}}$$

$$\frac{\Gamma \vdash_d \mathcal{A} \rightarrowtail (\{c_0, \ldots, c_n\},\ \{\ a_0, \ldots, a_n\})}{\Gamma \vdash_i \mathcal{A} \rightarrowtail \quad \begin{array}{l}\text{\%\{} \\ \quad \text{action: show(}[a_0, \ldots, a_n]\text{)} \\ \quad \text{condition: } c_0 \text{ and } \ldots \text{ and } c_n \\ \quad \text{state: Map.merge(}a_0\text{, Map.merge(}\ldots\text{, } a_n\text{))} \\ \text{\},}\end{array}} \;\texttt{(INFO-DEF)}$$

$$\frac{\begin{array}{c}\Gamma \vdash_v v \rightarrowtail x \\ \Gamma \vdash_i \mathcal{A}_0 \rightarrowtail \texttt{i}_0 \\ \vdots \\ \Gamma \vdash_i \mathcal{A}_n \rightarrowtail \texttt{i}_n\end{array}}{\Gamma \vdash_d \exists I \in v : \mathcal{A}_0 \vee \cdots \vee \mathcal{A}_n \rightarrowtail \quad \begin{array}{l}\text{Enum.map(x, fn (i) -¿ [} \\ \qquad \texttt{i}_0\text{,} \\ \qquad \texttt{...} \\ \qquad \texttt{i}_n \\ \quad \texttt{]} \\ \texttt{end}\end{array}} \;\texttt{(INFO-EX)}$$