1 Abstraction of BAKERY(3) by TLV

Task 1: Use TLV to compute predicate abstraction for BAKERY(3) and model check the properties of mutual exclusion and accessibility for the abstracted program. You can follow the class presentation for BAKERY(2) and use the files abs-bakeryN.smv and abs-baskeryN pf as a basis for this assignment. These files are available on the web page. Please remember to change the definition of \( N \) to \( N := 3 \), and extend the predicate base to cover the variables of all 3 processes.

2 Adequate and Inadequate Abstractions

Consider program simple, presented in Fig. 1. The property we wish to establish for this

\[
\begin{align*}
\forall x, y : \text{integer} \\
\ell_0: & \quad y := x + 1 \\
\ell_1: & \quad \text{while } x \neq 0 \text{ do} \\
& \quad \ell_2: \quad y := x \\
& \quad \ell_3: \quad x := x - 1 \\
& \quad \ell_4: \quad y := y - 1 \\
\ell_5: & \quad \text{end while}
\end{align*}
\]

Figure 1: Program simple.

program is the invariance of the assertion

\[ \varphi : \quad \text{at } \ell_5 \rightarrow y = 0 \]
Task 2: Use TLV in order to compute an abstraction program SIMPLE, where the abstraction mapping is given by

\[ \alpha: \quad X = \text{sign}(x), \quad Y = \text{sign}(y). \]

In order to do so, prepare first an SMV file containing the concrete version of this program. In this concrete program, it is enough to declare the variables \( x \) and \( y \) as ranging over the domain \([-5, 5]\). Make all transitions attempting to subtract 1 from \(-5\) disabled. That is, do not allow such transitions, which may cause the program to get stuck in front of some subtraction statement.

Then, in the corresponding \texttt{simple.pf} file, define the abstraction mapping by an appropriate conditional expression. For example, the mapping \( X = \text{sign}(x) \) can be expressed by the SMV assertion

\[
\begin{align*}
X = \text{case} \\
&\quad x < 0 : -1; \\
&\quad x = 0 : 0; \\
&\quad 1 : 1; \\
&\quad \text{esac}
\end{align*}
\]

Check whether the abstracted version of program SIMPLE satisfies the abstracted version of assertion \( \varphi \). Use TLV to compute this abstracted version.

Task 3: Repeat the above exercise with a more refined abstraction, defined by the following SMV assertion (for both \( X \) and \( Y \)):

\[
\begin{align*}
X = \text{case} \\
&\quad x < 0 : -1; \\
&\quad x = 0 : 0; \\
&\quad x = 1 : 1; \\
&\quad 1 : 2; \\
&\quad \text{esac}
\end{align*}
\]

Please submit your assignment electronically as a “postscript”, PDF, or “Word” file. Submit any *.smv or *.pf files as separate attachments. Preferably bundle all of the submitted files into a single “tar” or “zip” file.