Howto Typeset UPPAAL in LaTeX

Marius Mikučionis

December 19, 2008

Listings

We propose to use `listings` package to include UPPAAL declarations in LaTeX documents. `listings.sty` is distributed with `texlive-latex-recommended` Debian GNU/Linux package. `lstlisting` has many more options like colors, frames, columns, see documentation (install `texlive-latex-recommended-doc` on Debian):
http://www.ctan.org/get/macros/latex/contrib/listings/listings.pdf

Listing 1 shows UPPAAL declarations with syntax highlighted using bold and italic, while Listing 2 shows how declarations would look like in UPPAAL GUI.

```
1  const int N = 5;
2  typedef scalar[N] id_t;
3  int[0, 4] top[id_t];
4  bool yes100 = true;
5  meta int p := 20;
6  clock c; // single comment line
7  urgent broadcast chan ASAP;
8  typedef struct {
9      int one;
10     bool two;
11 } blob_t;
12 /**********************************
13  * It's problably not a good example,
14  * but good enough for LaTeX demo.
15  */
16 void reset(int v) {
17     if (0 <= v < N)
```

```
18            for (i: id_t) top[i] = v;
19        for (i: int[0,4])
20            if (i==3) continue;
21            else v++;
22        switch (v) { // bug#451
23            case 0: p++;
24            case 1: p++; break;
25            case 2: p--; break;
26            default: return;
27        }
28    }
29    before_update { reset(1) }
30    after_update { reset(2) }
31    system Process;
32    progress { p; }
```

Listing 1: UPPAAL code with column alignment, source in Listing 4.

If you don't like the broken frames across several pages, you can use parameter `float` with arguments similar to `figure`, see Listing 7.

```
const int N = 5;
typedef scalar [N] id_t ;
int [0, 4] top[id_t ];
bool yes100 = true;
meta int p := 20;
clock c; // single comment line
urgent broadcast chan ASAP;
typedef struct {
    int one;
    bool two;
} blob_t ;
/************************************
 * It 's problably not a good example,
 * but good enough for LaTeX demo.
 */
void reset (int v) {
    if (0 <= v < N)
        for (i: id_t ) top[i] = v;
    for (i: int [0,4])
        if (i==3) continue;
        else v++;
    switch (v) { // bug#451
        case 0: p++;
        case 1: p++; break;
        case 2: p--; break;
        default: return;
    }
}
before_update { reset(1) }
after_update { reset (2) }
system Process;
progress { p; }
```

Listing 2: UPPAAL code with colors, w/o column alignment, source in Listing 5.

In order to get things done, `listings` needs two things: the language description and the listing in raw text. Listing 3 shows UPPAAL declaration language description in three flavors that one can copy-paste to a LaTeX document. Listing 4 shows how to include UPPAAL declarations from external text file and Listing 5 shows how to typeset UPPAAL declarations within LaTeX file.

Listing 3: LaTeX code to be placed in document preamble.

```
\usepackage{latexsym,multicol,color,pstricks}
\usepackage{listings} % comes with texlive-latex-recomended

\lstdefinelanguage{Uppaal}{ % syntax highlight via font
  basicstyle=\small\sffamily, % small sans-serif font (like verdana)
  keywords={after_update,assign,before_update,break,case,const,continue,
    default,else,enum,for,guard,if,meta,process,progress,return,select,
    state,sync,switch,trans,system,while},
  keywords={[2]broadcast,bool,clock,chan,commit,init,int,scalar,struct,
    typedef,urgent,void}, keywordstyle={[2]\bfseries},
  keywords={[3]false,true}, otherkeywords={[3]->},
  morekeywords={[3]->}, keywordstyle={[3]\bfseries},
  comment=[l]{//}, morecomment=[s]{/*}{*/}, % single and multi-line
  commentstyle=\itshape, % appear in italic
  tabsize=4, % tab treatment (going to be fixed in Uppaal)
  captionpos=b, % put captions at the bottom
  escapechar=@ % write LaTeX comments escaped by @ symbol
}

\lstdefinelanguage[GUI]{Uppaal}[]{Uppaal}{ % syntax like in GUI
  keywordstyle={[2]\color{black!50!green}}, % slightly darker than in GUI
  otherkeywords={->}, keywordstyle={[3]\color{magenta}},
  commentstyle={\color{black!50!red}\itshape}, % dark red
  literate={{->}{$->$}3} % fix arrows
}

\lstdefinelanguage[LIT]{Uppaal}[GUI]{Uppaal}{ % replace some symbols
  literate={{->}{{$\leadsto$} }2 {-->}{{$\longrightarrow$} }2
    {=}{{$\gets$ }}2 {==}{{$==$}}2 {:=}{{$\gets$ }}2 {<=}{{$\leq$ }}2
    {>=}{{$\geq$ }}2 {&&}{{$\land$}}2 {||}{{$\lor$}}2 {<>}{{$\Diamond$}}1
    {[]}{{$\Box$}}1 {forall}{{$\forall$}}1 {exists}{{$\exists$}}1}
}
```

Note that column alignment is enforced by default. This may look better but it also breaks copy-paste feature from PDF. We use [columns=flexible] to remove the alignment effort which allows user to copy-paste clean code like in the following LaTeX listings, cheers!

Listing 4: LaTeX source of Listing 1.

```
\lstinputlisting[language={Uppaal}, % use simple Uppaal flavor
    numbers=left,numberstyle=\tiny,stepnumber=1,numbersep=10pt,xleftmargin=7mm,
    firstnumber=1,frameround=fttt, frame=trBL, % put rounded frame
    caption={{\sc Uppaal} code with column alignment, source in
      Listing~\ref{lst:blacksource}.}, firstline=85, lastline=116,
    label=lst:black]{uppaal-listing.tex}
```

Listing 5: LaTeX source of Listing 2.

```
\begin{lstlisting}[language={[GUI]Uppaal}, % use GUI flavor
    columns={[l]flexible},
    frameround=fftt, frame=shadowbox, rulesepcolor=\color{gray},
    caption={{\sc Uppaal} code with colors, w/o column alignment, source
      in Listing~\ref{lst:guisource}.}, label=lst:gui]
const int N = 5;
typedef scalar[N] id_t;
int[0, 4] top[id_t];
bool yes100 = true;
meta int p := 20;
clock c; // single comment line
urgent broadcast chan ASAP;
typedef struct {
    int one;
    bool two;
} blob_t;
/**********************************
 * It's problably not a good example,
 * but good enough for @\LaTeX@ demo.
 */
void reset(int v) {
    if (0 <= v < N)
        for (i: id_t) top[i] = v;
    for (i: int[0,4])
        if (i==3) continue;
        else v++;
    switch (v) { // bug#451
        case 0: p++;
        case 1: p++; break;
        case 2: p--; break;
        default: return;
    }
}
before_update { reset(1) }
after_update { reset(2) }
system Process;
progress { p; }
\end{lstlisting}
```

It is also possible to make listings in multiple columns using `multicol` package and adding [multicols=2] option, but it does not look good if caption and frame is used, see Listing 6, which is an example of "dump everything in text".

```
const int N = 5;
typedef scalar [N] id_t;
int [0, 4] top[id_t];
bool yes100 = true;
meta int p := 20;
clock c; // single comment line
urgent broadcast chan ASAP;
typedef struct {
    int one;
    bool two;
} blob_t;
/************************************
 * It's problably not a good example,
 * but good enough for LaTeX demo.
 */
void reset(int v) {
    if (0 <= v && N-1 >= v)
        for (i: id_t) top[i]:=v;
    for (i: int [0,4])
        if (i==3) continue;
        else v++;
    switch (v) { // bug#451
        case 0: p++;
        case 1: p++; break;
        case 2: p--; break;
        default: return;
    }
}
before_update { reset(1) }
after_update { reset (2) }

process Template() {

state
    loc2Name {inv2Label},
    locName {invariantLabel};
commit
    loc2Name;
urgent
    locName;
init locName;
trans
    loc2Name -> locName { },
    locName -> loc2Name { select
        selectLabel; guard guardLabel;
        sync syncLabel; assign
        updateLabel; };
}
// Place template instantiations here.
Process = Template();
// List one or more processes to be
    composed into a system.
// Try -- >, E<> and A[] properties
system Process;
progress { p; }
```

Listing 6: UPPAAL XTA file.

Listing 7: LaTeX source of Listing 6 header.

```
\setlength{\columnsep}{25pt} % line numbers tend to overlap
\begin{lstlisting}[float=htb,language={[GUI]Uppaal}, % use GUI flavor
  columns={[l]flexible}, multicols=2,
  frameround=fftt, frame=shadowbox, rulesepcolor=\color{gray},
  numbers=left,numberstyle=\tiny,stepnumber=1,numbersep=5pt,
  breaklines=true, breakatwhitespace=true, % enable line breaks
  caption={{\sc Uppaal} XTA file.}, label=lst:xta]
```

Listing 8 is similar to Listing 6 except that some symbols are replaced by math typesetting, this is so called "literate" feature of `listings`.

Listing 8: "Literate" XTA file.

```
const int N ← 5;
typedef scalar [N] id_t ;
int [0, 4] top[id_t ];
bool yes100 ← true;
meta int p ← 20;
clock c; // single comment line
urgent broadcast chan ASAP;
typedef struct {
    int one;
    bool two;
} blob_t ;
/************************************
 * It 's problably not a good example,
 * but good enough for LaTeX demo.
 */
void reset (int v) {
    if (0 ≤ v ∧ N−1 ≥ v)
        for (i : id_t ) top[i] ← v;
    for (i : int [0,4])
        if (i ==3) continue;
        else v++;
    switch (v) { // bug#451
        case 0: p++;
        case 1: p++; break;
        case 2: p−−; break;
        default: return;
    }
}
before_update { reset(1) }
after_update { reset (2) }

process Template() {

state
    loc2Name {inv2Label},
    locName {invariantLabel};
commit
    loc2Name;
urgent
    locName;
init locName;
trans
    loc2Name ⇝ locName { },
    locName ⇝ loc2Name { select
        selectLabel; guard guardLabel;
        sync syncLabel; assign
        updateLabel; };
}
// Place template instantiations here.
Process ← Template();
// List one or more processes to be
    composed into a system.
// Try ⟶ , E◇ and A□ properties
system Process;
progress { p; }
```

Listing 9: LaTeX source of Listing 8.

```
\setlength{\columnsep}{25pt} % line numbers tend to overlap
\lstinputlisting[language={[LIT]Uppaal}, % use LIT flavor
  columns={[l]flexible}, multicols=2, captionpos=t,
  numbers=left,numberstyle=\tiny,stepnumber=1,numbersep=5pt,
  breaklines=true, breakatwhitespace=true, % enable line breaks
  caption={"Literate" XTA file.}, label=lst:xtaliterate,
  firstline=166, lastline=216, firstnumber=1]{uppaal−listing.tex}
```

Listing 10: `listings` language description for LaTeX syntax.

```
\lstdefinelanguage[listings]{TeX}[LaTeX]{TeX}{morekeywords={\sc,\color},
  keywords={[2]\lstdefinelanguage,\lstinputlisting,columns,basicstyle,tabsize,
    emph,emphstyle,morekeywords,lstlisting,language,numbers,numberstyle,
    stepnumber,numbersep,caption,label,frameround,frame,multicols,keywords,
    comment,morecomment,commentstyle,breaklines,breakatwhitespace,rulesepcolor,
    setlength,escapechar,otherkeywords,keywordstyle,literate,captionpos,
    firstline,lastline,firstnumber,xleftmargin,xrightmargin,float},
  keywordstyle={[2]\color{blue}}, morecomment=[l][\color{red}\itshape]{\%},
  keywords={[3]Uppaal,GUI,LIT}, keywordstyle={[3]\color{black!50!green}},
  columns=fullflexible}
```