

On free variables in process specifications, LPSs and PBESs

Jan Friso Groote

March 2, 2009

In process specifications and in parameterised boolean equation systems (PBESs), it is possible to use free variables. An example is the following:

```
act     $a$ ;  
glob   $x : \mathbb{N}$ ;  
proc   $P = a(x).P$ ;  
init   $P$ ;
```

This represents a whole class of processes, namely for every value of x this process has a different value. The keyword **glob** stands for *global variable*¹. In each specification the keyword **glob** can be used once any arbitrary of times. The scope of the variables mentioned in a **glob** declaration is the global file. All **glob** declarations are grouped together. The names of the variables cannot coincide with other declared functions, processes, actions, variables (both in equations and in sum operators) and process and PBES parameters. Global variables can occur in process equations, in parameterised fixed point formulas and in **init** sections.

Such global variables can be used in the common mathematical way. Consider for instance the polynomial:

$$ax^2 + bx + c = 0.$$

There are four variables in this equation, namely a , b , c and x . The use of the variables a , b and c allow to study this polynomial in a far more general setting than when these variables would have concrete values.

In some cases, the concrete values for global variables do not have influence on the process. In such a case instantiating the global variables to various concrete values will mean that the process has the same behaviour modulo strong bisimulation, or the same solution as a parameterised boolean equation system. In this case we call the process or PBES *global variable insensitive*. An example is the following linearisation of the buffer

¹The keyword **glob** was not used up till now. Currently, **lpspp** prints the keyword **var** but this is confusing, as **var** in equations has a different use and meaning. It is however important that the keyword is short. **freevar** is too long.

```

sort    $D$ ;
glob    $dummy_1, dummy_2 : D$ ;
proc    $P(b : \mathbb{B}, d : D) = \sum_{e : D} b \rightarrow read(e) \cdot P(false, e) + \neg b \rightarrow send(d) \cdot P(true, dummy_1)$ ;
init    $P(true, dummy_2)$ ;

```

The idea is that if the parameter b is *true*, the value of the second parameter is not relevant anymore. Therefore, it can be set to any arbitrary value, which is indicated by the use of *dummy₁* and *dummy₂*. As this specification is global variable insensitive, concrete values can be chosen for these global variables when this would be fruitful.

The tool `mcr122lps` generates linear processes with global variables. It guarantees that the resulting specification is global variable insensitive. Certain transformation tools, like `lpsconstelm` and `lps2pbes` yield global variable insensitive output, provided the input is global variable insensitive. In case systems are not global variable insensitive the output of these tools can be garbage. It is the responsibility of those who apply the tools that the tools are used in a proper way. It is likely that most tools leave global variables untouched, unless a switch indicates that global variable insensitivity can be used.