Static verification with VerCors

Classical examples of software faults include the $400 million Pentium bug and the THERAC-25, which caused fatal injuries to numerous people.

Future-based Analysis of Message Passing Programs via static analysis and model checking

Wytse Oortwijn, Stefan Blom, and Marieke Huisman

Challenge

Statically proving functional correctness of message passing programs is very hard...

Message exchanges are often concurrent, the number of processes are often unknown, and the number of possible behaviours is often infinite.

MOTIVATION

...but having the ability to proof their correctness is crucial....

CONTRIBUTION

...we propose future-based analysis to verify the communicational aspects.

With the VerCors toolset we can show local correctness and capture communicational behaviour in abstract models, called futures, which we analyze to also prove global and functional correctness.

Example program

An example with 1 server and 3 clients. Each client sends an integer to the server. The server sums up and broadcasts the sum of all values. The futures are transformed into input for mCRL2, otherwise a fail verdict is reported.

Example future specification

The future annotations, which are mCRL2 process terms, describe the communication behaviour. For example, the following two processes correspond to the example program:

Transforming model checking results

The output of mCRL2 is analyzed and transformed to allow its projection onto the concrete source code.

Final verdict

The VerCors toolset either successfully verifies the program with the given future specification, or indicates a violation and gives a report.

Static verification + model checking

With mCRL2 we check for communication correctness, e.g. validity of message exchanges, resource leakage, and other interesting safety and liveness properties.