Testing the Temporal Behavior of Real-Time Software Modules using Extended Evolutionary Algorithms

Hartmut Pohlheim, Joachim Wegener
DaimlerChrysler AG, Research and Technology, Alt-Moabit 96a, 10559 Berlin, Germany
phone: +49 30 39982 {172, 232}, fax: 107, email: { pohlheim, wegener}@dbag.bln.daimlerbenz.com

Many industrial products are based on the use of embedded computer systems. Usually, these systems have to fulfill real–time requirements, and correct system functionality depends on their logical correctness as well as on their temporal correctness. Therefore, the developed systems have to be thoroughly tested in order to detect existing deficiencies in temporal behavior, as well as to strengthen the confidence in temporal correctness.

Existing test methods are not specialized in the examination of temporal correctness. For this reason, new test methods are required which concentrate on determining whether the system violates its specified timing constraints. Normally, a violation means that outputs are produced too early, or their computation takes too long. The task of the tester therefore is to find the input situations with the longest or shortest execution times, in order to check whether they produce a temporal error. It is virtually impossible to find such inputs by analyzing and testing the temporal behavior of complex systems manually. However, if the search for such inputs is interpreted as a problem of optimization, evolutionary computation can be used to find the inputs with the longest or shortest execution times.

We have developed and examined a new approach for testing temporal behaviour which is based on the use of evolutionary algorithms, namely evolutionary testing [2]. Our work investigates the effectiveness of Evolutionary Algorithms to validate the temporal correctness of embedded systems by establishing the maximum and minimum execution times.

Here we present results from one application field: a motor control system (table 1). It contains several modules which have to fulfill timing constraints. These modules were tested on the target processor later used in the vehicles. The execution times were measured by means of hardware timers.

The results of evolutionary testing were compared to the maximum execution times determined by the developers with systematic testing. These values are shown in the column labeled developer test.

Table 1: Maximum execution times of motor control software modules determined by evolutionary and systematic testing

module name	max. exec. time in µs: evolutionary testing	developer test	lines of code	parameters
zr2	69,6 µs	67,2 µs	41	18
t1	120,8 µs	108,4 µs	119	18
mc1	112,0 µs	108,4 µs	98	17
mr1	68,8 µs	64,0 µs	81	32
k1	59,6 µs	57,6 µs	39	14
zk1	58,4 µs	54,0 µs	56	9

A comparison of the results between developer test and evolutionary test showed that evolutionary testing found longer execution times for all the given modules. This is especially astonishing, because the Evolutionary Algorithm treats the software modules as black boxes.

For the optimization we employed extended Evolutionary Algorithms. This includes the use of multiple subpopulations, each using a different search strategy. Additionally, competition for limited resources between these subpopulations have taken place [1].

However, the use of Evolutionary Algorithms alone is not sufficient for a thorough and comprehensive test of real–time systems. A combination with existing test procedures is necessary to develop an effective test strategy for embedded systems which examines functional as well as temporal system behaviour. A combination of systematic testing and evolutionary testing is promising [2].

References

[1] *Pohlheim, H.*: Genetic and Evolutionary Algorithm Toolbox for use with Matlab - Documentation. http://www.geatbx.com/, 1994-1999.

[2] *Wegener, J., and Grochtmann, M.*: Verifying Timing Constraints of Real-Time Systems by Means of Evolutionary Testing. Real-Time Systems, 15, pp. 275-298, 1998.