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

Hartmut Pohlheim, Joachim Wegener, Harmen Sthamer, DaimlerChrysler AG, Research and Technology, FT3/SM, Berlin, Germany

1 Motivation

Costs for development and test of embedded systems

→ 50 % implementation, 30 % unit testing, 20 % system testing

⇒ 50 % of development costs spent for testing

Objectives of testing real-time systems

→ finding errors in functional behavior

→ finding errors in temporal behavior

→ building up confidence in the correct functioning of the test object by executing the system under test with selected inputs

2 Testing temporal behavior

Testing temporal system behavior

→ find violations of specified timing constraints (find the inputs with longest and shortest execution time)

→ check whether they produce a temporal error (outputs are produced prematurely or their computation takes too long)

⇒ testing temporal behavior is a complex task

Problem:

→ lack of appropriate test procedures

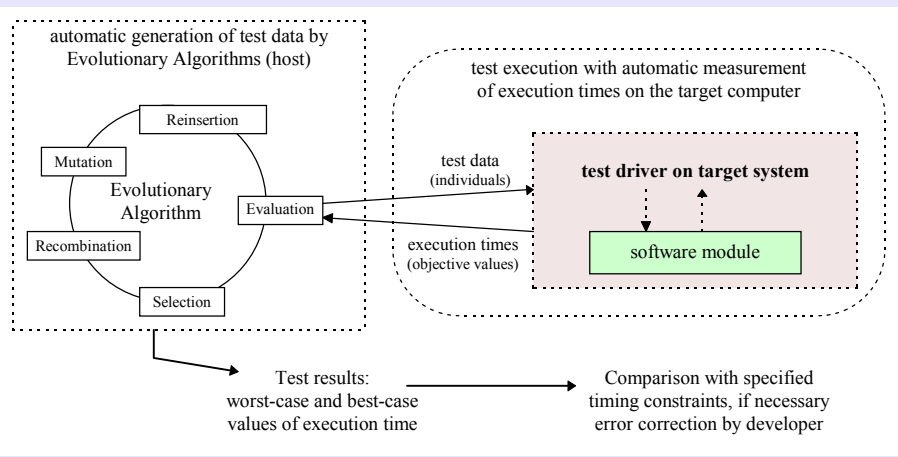
→ tester uses conventional test methods

Evolutionary Testing

Use of Evolutionary Optimization to determine longest and shortest execution times automatically.

3 Evolutionary Testing

Integration of Evolutionary Algorithm and time measurement equipment



- each individual represents a test datum
- execution time for each test datum determines its objective value
- when looking for worst-case execution time: test data with long execution time obtain high fitness values and vice versa

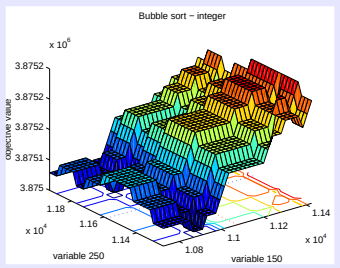
employed tools:

- TESSY by DaimlerChrysler AG, (ATS GmbH, www.ats-berlin.de)
- GEATbx: Genetic and Evolutionary Algorithm Toolbox for Matlab (www.geatbx.com)

4 Search space analysis

- temporal behavior forms a very complex multi-dimensional search space
- many plateaus, local minima and discontinuities
  - several input data executing the same program path are identical
  - different program paths lead to irregular changes of the exec. times
- two analysis examples provided
  - bubblesort modul (benchmark)
  - feature extraction ME (real world)

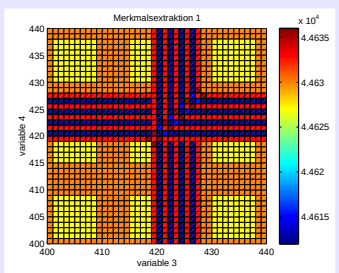
Analysis of bubblesort modul



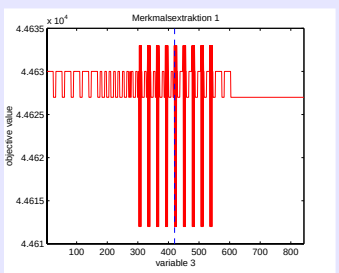
3-D variation plot (execution time of bubblesort modul): variable 150 and 250 in a small area (0.6%) of the search space

- list of 500 variables, each variable in range [-32768, 32767]
- bubble sort variables
- analysis results:
  - many plateaus
  - many local minima
  - correlation between variables

Analysis of feature extraction modul (ME)



2-D color quilt variation plot (execution time of feature extraction): toggle brightness (min/max variable value) at 2 variable positions (400-440)



2-D variation plot (execution time of feature extraction): toggle 1 brightness value over all variable positions

- 843 variables in [0, 4095]
- defining a 29x29 pixel matrix out of an 287x1200 pixel picture
- analysis results:
  - large steps in objective values even for small changes in variable values
  - large plateaus without change of objective value

5 Experiments

Optimization of engine control software modules

1st industrial application of Evolutionary Testing

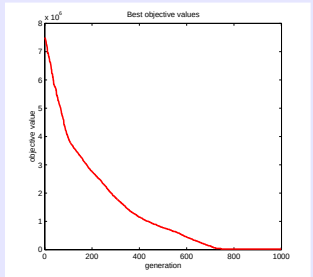
- results were compared to the times determined by the developers with systematic testing and random testing

- found longest execution times for all given modules
- used considerable less time (1/3 of random testing)
- higher reliability than random testing

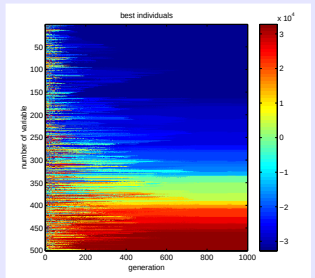
- Evolutionary Algorithm for integer variables:
  - 3 subpopulations each 20 individuals over 100 generations
  - different strategies per subpopulation, competition
  - discrete recombination and integer mutation (different range)

	max. execution time in µs				
module name	Evolutionary Testing	developer test	random testing	program lines	para-meters
zr2	**69,6 µs**	67,2 µs	66,4 µs	41	18
t1	**120,8 µs**	108,4 µs	116,0 µs	119	18
mc1	**112,0 µs**	108,4 µs	110,0 µs	98	17
mr1	**68,8 µs**	64,0 µs	45,2 µs	81	32
k1	**59,6 µs**	57,6 µs	54,0 µs	39	14
zk1	**58,4 µs**	54,0 µs	**58,4 µs**	56	9

Optimization of bubblesort modul



Optimization of bubblesort modul (minimal execution time): best objective value over 1000 generations, optimum found in generation 746



Optimization of bubblesort modul (same run as left): variables of best individual per generation over 1000 generations ("sorting of variables")

- benchmark problem for execution time measurement
- Evolutionary Algorithm for integer/permutation variables:
  - 6 subpopulations (regional model), each 50 indiv., 1000 generations
  - different strategies per subpopulation
  - discrete recombination / order crossover
  - integer mutation and swap mutation
  - competition between subpopulations

6 Summary

New approach:

Testing temporal behavior by Evolutionary Testing

Advantages:

+ automatic search for the longest and shortest execution times
+ suited for discontinuous target functions
+ suited for complex input domains with many parameters
+ can escape local optima (search by multiple individuals)
+ direct assessment of objective value: execution time of individual

But:

– finding the extreme execution times is **not** guaranteed

- Combination of systematic testing and Evolutionary Optimization opens up further potential for improvement
  - → perform systematic test to examine functional correctness
  - → use systematically produced test data to inoculate initial population of Evolutionary Algorithm
  - → apply Evolutionary Testing to find extreme execution times

References

[1] *Grochtmann, M., and Wegener, J.*: Evolutionary Testing of Temporal Correctness. Proceedings of Quality Week Europe '98, 1998.

[2] *Pohlheim, H.*: Evolutionäre Algorithmen - Verfahren. Operatoren, Hinweise aus der Praxis. Berlin, Heidelberg, Germany: Springer-Verlag, 1999. http://www.pohlheim.com/eavoh/index.html

[3] *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.

[4] *Wegener, J., and Pitschinetz, R.*: TESSY - Yet Another Computer–Aided Software Testing Tool? Proceedings of the Second European International Conference on Software Testing, Analysis & Review EuroSTAR '94, 1994.

Pohlheim, May 2000