KTH VETENSKAP OCH KONST

KTH Computer Science and Communication

LBT for Procedural and Reactive Systems

Part 4: Procedural Systems

Karl Meinke,

karlm@kth.se

KTH Royal Institute of Technology Stockholm

0. Overview of Talk

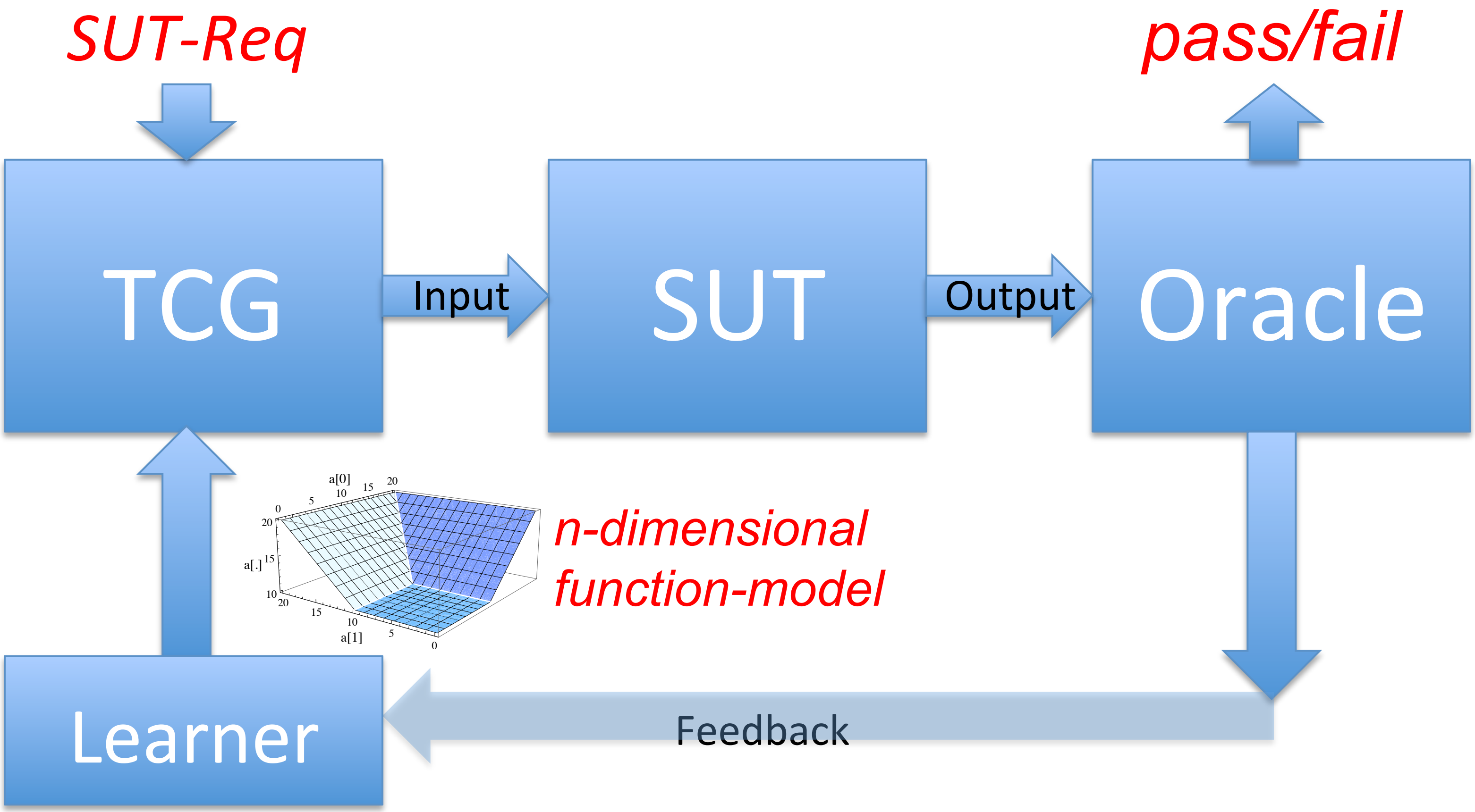
1. Introduction and Motivation
2. Technical Approach
3. Benchmarking Results
4. Conclusions

Based on:

K. Meinke, *Automated Black-Box Testing of Functional Correctness using Function Approximation*, in: Proc ISSTA 2004

K. Meinke and F. Niu, *A Learning-Based Approach to Unit Testing of Numerical Software*, in Proc. ICTSS 2010

1.2. LBT for Procedural Code



Q: How would MBT cope with this scenario?

1.3. LBT for Numerical Codes

- Requirements Language – Hoare logic over
  - first-order logic over real-closed fields
- Models
  - non-gridded n-dimensional piecewise polynomials of low degree (d=1,2,3) (= "n-wise testing")
- Model checker
  - Hoon-Collins CAD algorithm, a satisfiability algorithm (Mathematica)
- Learning algorithm
  - local polynomial interpolation

1.4 Why Numerical Code?

- M.G. Cox, P.M. Harris, E.G. Johnson, P.D. Kenward, and G.I. Parkin. *Testing the numerical correctness of software*. Technical Report CMSC 34/04, National Physical Laboratory, Teddington, January 2004.

- Showed numerical errors in NAG, IMSL, Microsoft Excel, Lab-VIEW and Matlab,

- Numerical specifications exist!
- Insight into other cases (e.g. integers)
- Data-oriented testing
- The algorithms and models fit together extremely well!

Technical Approach

2.1 Models of Numerical Code

- Assume numerical code can be modeled as a function $f : \Re^m \rightarrow \Re^n$
- (ignores non-termination)
- Decompose into *n* co-ordinate functions

$f_i : \Re^m \rightarrow \Re, \quad i = 1, \dots, n$

2.2 Medial Spheres Approximation

Decompose $f_i : \Re^m \rightarrow \Re$ into piecewise polynomial approximations $f_i^1, \ldots, f_i^k$

over m-dimensional spheres $S_1, \ldots, S_k$ with centres $c_1, \ldots, c_k$, and radii $r_1, \ldots, r_k$

Looks like *Weierstrass Theorem* on polynomial approximation.

Piecewise methods tolerate discontinuities.

Correct approximation theory is *medial sphere approximation* (c.f. solid modeling)

2.3 Approximation Support

- $(d+1)^m$ points in S needed to uniquely determine an m-dimensional degree d interpolating polynomial $p(S)$

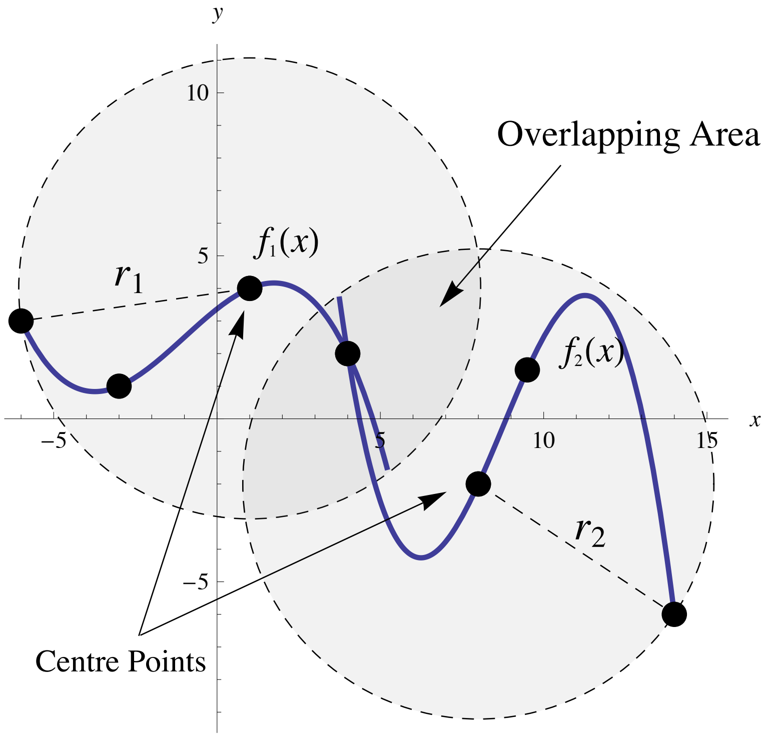
- $x_1, f(x_1), \ldots, x_{(d+1)^m}, f(x_{(d+1)^m})$

- $x_1, , \ldots, x_{(d+1)^m}$ are the support for interpolant $p(S)$
- Point can be randomly placed in S.
- $p(S) \neq 0$ tends to infinity outside S,
  - so no extrapolation!
- Spheres S_i can (and should) overlap for smoothness

2.4 Choosing Support Sets

- A gridded approach to data sampling doesn't work
- Exponential blowup in grid-point number with dimension size (m)
- Can't sample off-grid so might miss bugs
- Need a non-gridded approach

2.5 Non-gridded Piecewise Models



Two overlapping 2-dimensional local models (cubics)

2.6 Model Refinement

- Every data point c becomes the centre of a sphere S_c
- Members of S_c are the $(d+1)^m - 1$ nearest members.
- As new points are added globally, spheres tend to shrink, improving their approximation accuracy.

2.7 Model Convergence

- Measure convergence of each sphere locally as an integral
- $\int_{Snew} p(S_{old}) - \int_{Snew} p(S_{new})$

- Compute this by a quick and dirty Monte Carlo approximation
- Choose least converged sphere as a breadth first search heuristic - minimise uncertainty

2.8 Requirements Modeling

- Use Hoare triples pre{code}post
- pre and post are arbitrary first-order formulas (quantifiable!) over language $L(\mathfrak{R})$ of real-closed fields.
- Tarski's Theorem "$Th_{L(\mathfrak{R})}(\mathfrak{R})$ is decidable"
- Hoon-Collins CAD algorithm
- *Cylindric algebraic decomposition*
- Doubly exponential time algorithm!
- Solvable for 6-8 free variables in practise.

2.9(a) What to solve?

- pre contains *invars* $x_1, \ldots, x_m$
- post contains $x_1, \ldots, x_m$ and *outvars* $x'_1, \ldots, x'_m$
- x' is post execution state of x, e.g.

$$x \geq 0.0 \; \{\text{Newton-code}\} \; | \, x'*x' - x \, | \leq \varepsilon$$

Replace x'_i by $p(S^i)\,(x_1, \ldots, x_m)$ in post, e.g.

$$x \geq 0.0 \; \{\text{Newton-code}\} \; | \, p(S^i)\,(x) * p(S^i)\,(x) - x \, | \leq \varepsilon$$

for each sphere S^i for co-ordinate f_i

2.9(b) What to Solve?

Solve for $x_1 ,\ldots, x_m \in \Re$ the formula

$$\mathrm{pre}(x_1 ,\ldots, x_m) \;\; \&$$
$$<x_1 ,\ldots, x_m> \in S^i ,\ i=1,\ldots,n \;\&$$
$$\neg\, \mathrm{post}(x_1 ,\ldots, x_{m,}\ p(S^1)\ (x_1 ,\ldots, x_m) ,\ \ldots,\ p(S^n)\ (x_1 ,\ldots, x_m)\ )$$

We call CAD on this formula, and can ask for several solutions to $x_1 ,\ldots, x_m$.

Use k-*wise testing* for large m, where $k < m$

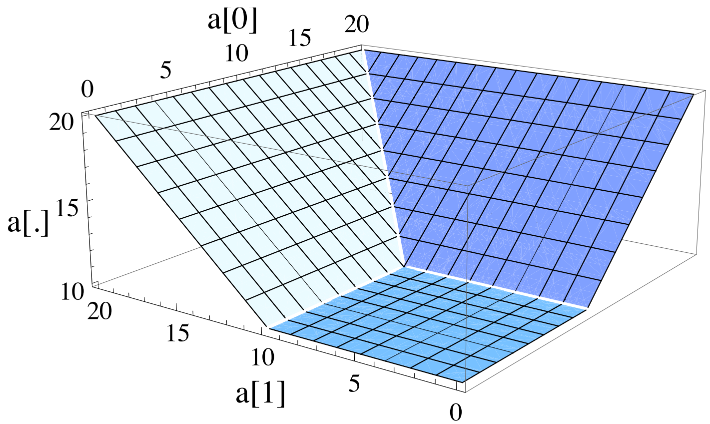
Part 3: Benchmarking Results

3.1 How to evaluate?

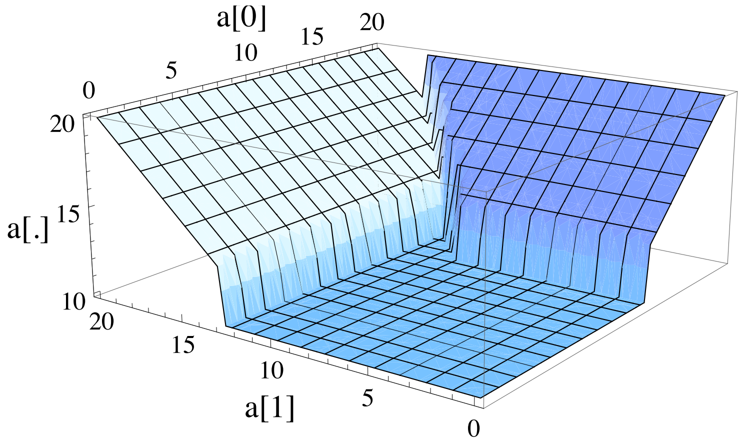
- Decided to benchmark against random testing.
- Small numerical algorithms are VERY fragile against mutations.
- Small mutation has large destructive effect.
- Built a random use-case generator
  - Randomly generated numerical functions
  - Associated formal specifications

3.4. Specific Case Studies

e.g. Bubblesort: LBT is 10X faster than random

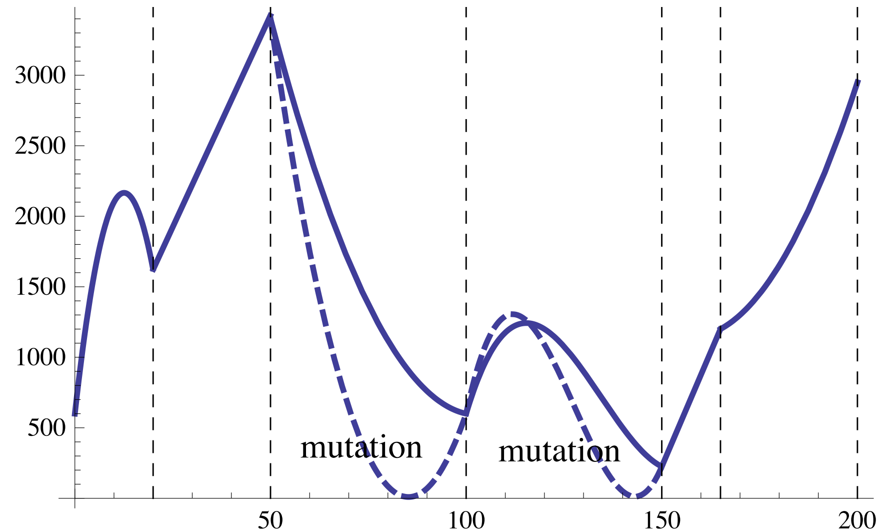


Model of
unmutated code



Model of
mutated code

3.2. Statistical Evaluation



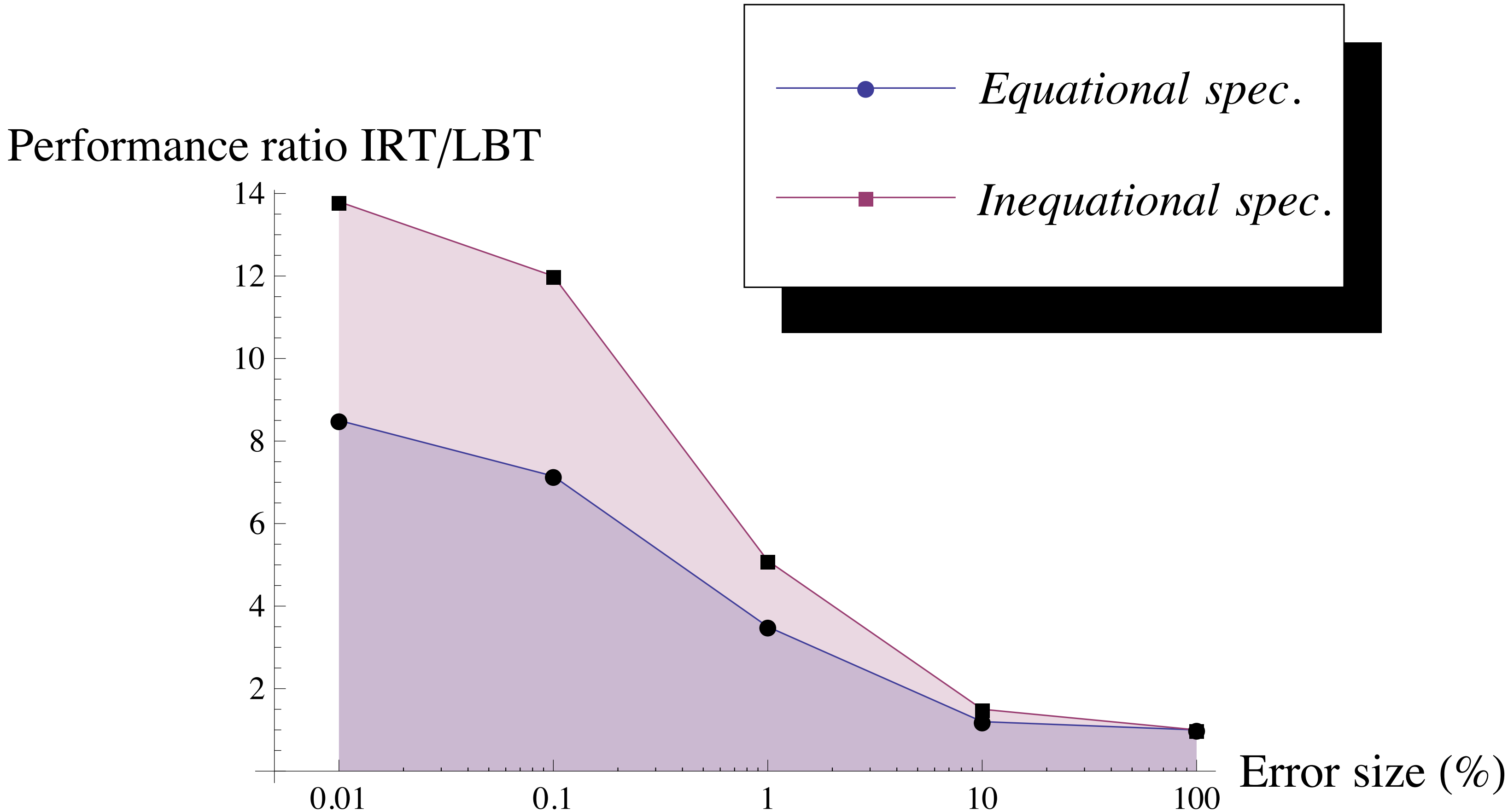
Randomly generated and mutated SUT

equations

inequalities

Automatically generated pre and postconditions

3.3. Statistical Benchmarking against Random TCG



4. Conclusions

- Computationally tractable case
- Good example of the LBT paradigm
- Interpolation "works" as inductive inference, especially due to continuity over $\Re$
- Convincing benchmark results
- Provides insight into data-oriented LBT
- Used these methods to learn *hybrid automata*

Open Questions

- ***N*** and ***Z*** are a whole different ball-game …

- Thanks to the HSST Organisors!