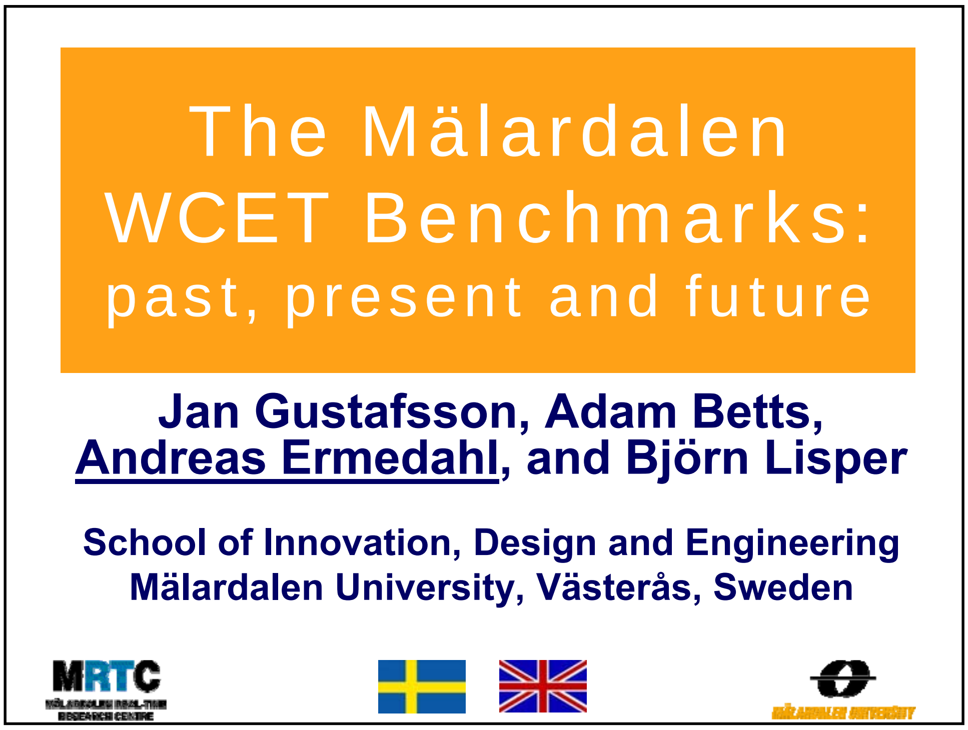
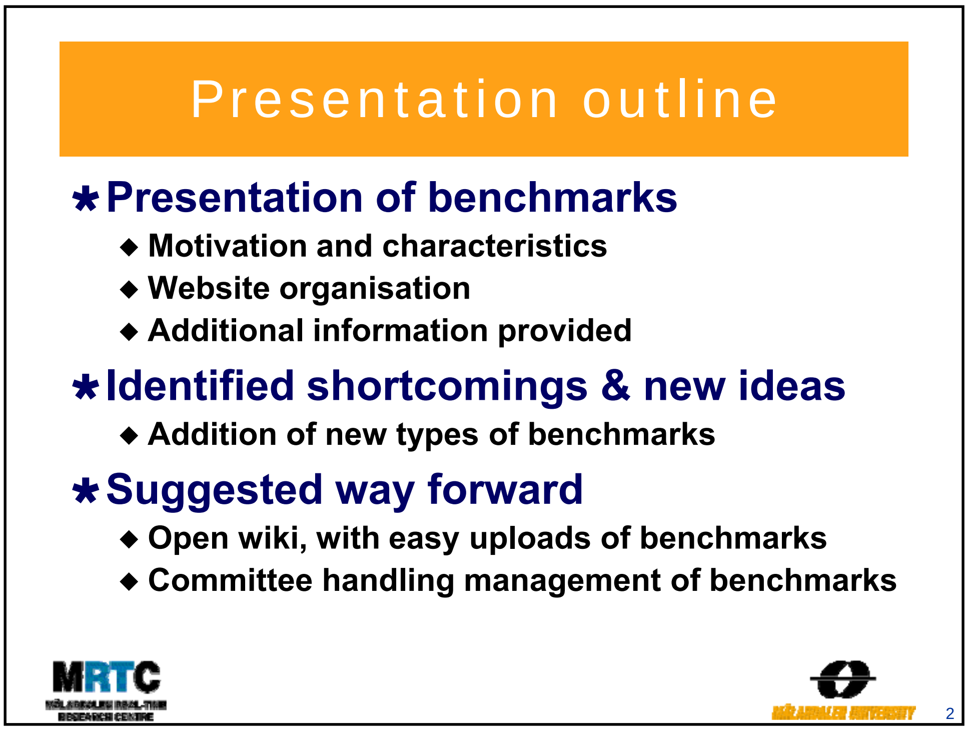


The Mälardalen WCET Benchmarks: past, present and future

Jan Gustafsson, Adam Betts, Andreas Ermedahl, and Björn Lisper

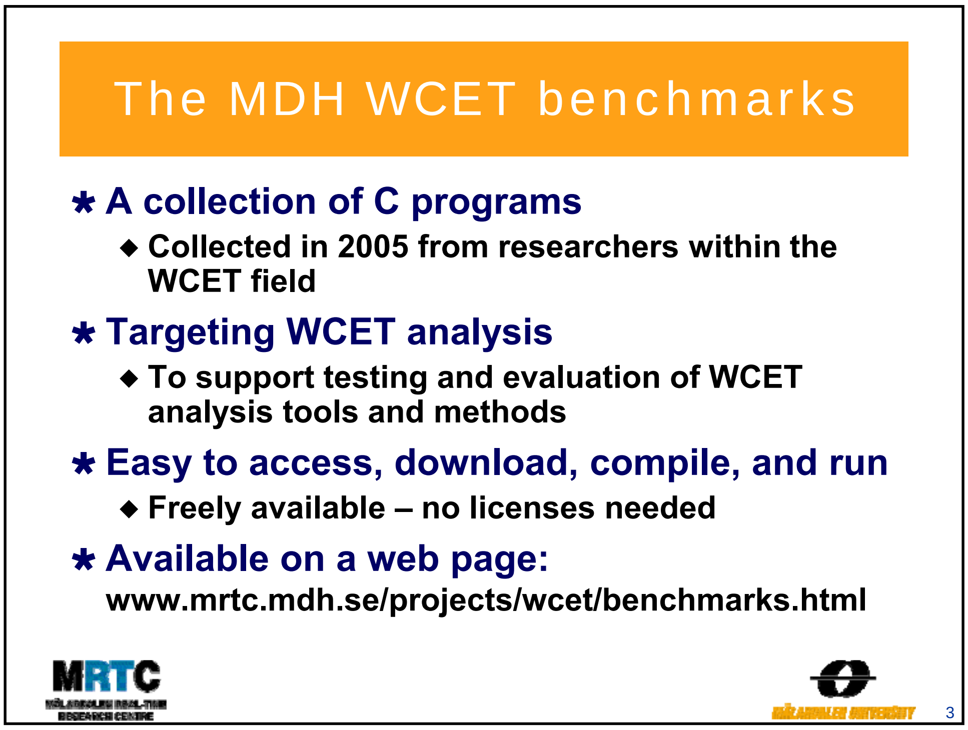
School of Innovation, Design and Engineering
Mälardalen University, Västerås, Sweden



Presentation outline

- Presentation of benchmarks
  - Motivation and characteristics
  - Website organisation
  - Additional information provided
- Identified shortcomings & new ideas
  - Addition of new types of benchmarks
- Suggested way forward
  - Open wiki, with easy uploads of benchmarks
  - Committee handling management of benchmarks

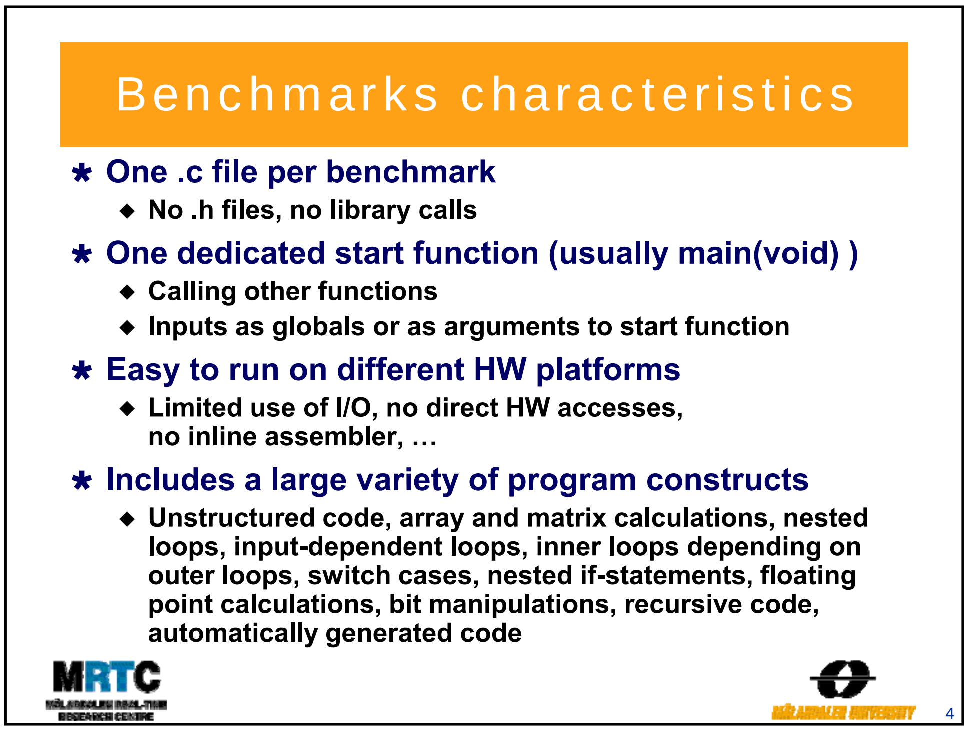
2



The MDH WCET benchmarks

- A collection of C programs
  - Collected in 2005 from researchers within the WCET field
- Targeting WCET analysis
  - To support testing and evaluation of WCET analysis tools and methods
- Easy to access, download, compile, and run
  - Freely available – no licenses needed
- Available on a web page:
  www.mrtc.mdh.se/projects/wcet/benchmarks.html

3



Benchmarks characteristics

- One .c file per benchmark
  - No .h files, no library calls
- One dedicated start function (usually main(void))
  - Calling other functions
  - Inputs as globals or as arguments to start function
- Easy to run on different HW platforms
  - Limited use of I/O, no direct HW accesses, no inline assembler, …
- Includes a large variety of program constructs
  - Unstructured code, array and matrix calculations, nested loops, input-dependent loops, inner loops depending on outer loops, switch cases, nested if-statements, floating point calculations, bit manipulations, recursive code, automatically generated code

4

Input value annotations

- **All benchmark contain their own input, and can run "as is"**
  - Single path programs ⇒ WCET analysis easy, just run once
  - Not realistic ⇒ Most embedded programs are input-dependant
- **Examples of real-world inputs:**
  - Environmental inputs using ports or memory mapped I/O
  - Parameters to main() or to function that invokes the task
  - Static variables keeping state of task between invocations
  - Task communication, e.g. global memory or message queues
- **Some benchmark have input value annotations**

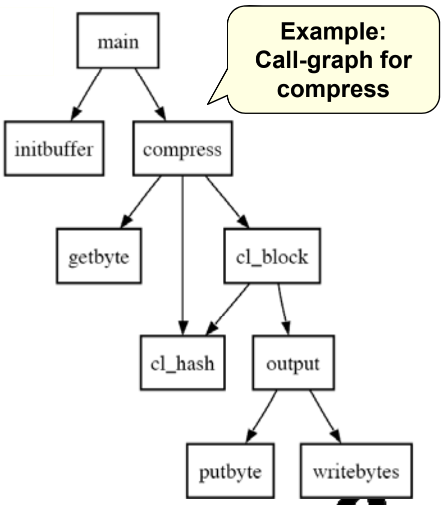
```
/* At entry to the call to complex: a=[0..18] b=[0..18] */
FUNC_ENTRY complex ASSIGN a INT 0 18 || b INT 0 18;
```

  - Intervals hold possible values of inputs at certain program points
  - Stored as .ann files at web-site
  - Only few programs, most benchmarks are single path

5

Provided graphs

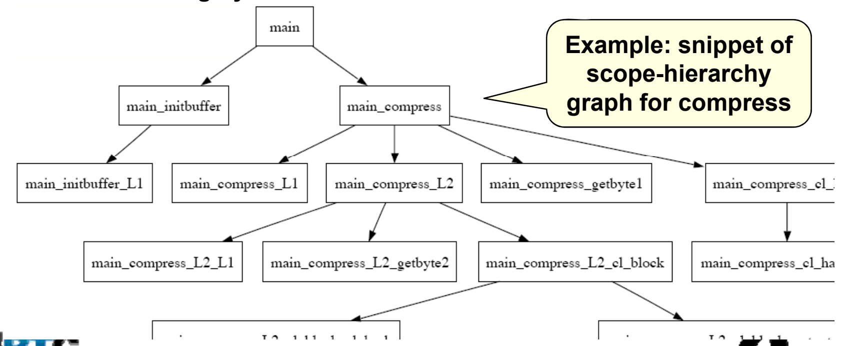
- **Call-graph**
  - Shows how different functions may call each other
  - Provided as a .pdf file



6

Provided graphs

- Scope-hierarchy graph
  - Context-sensitive graph
  - Scopes are functions and loops (each is given an unique name)
  - Each call-site creates scope(s) of the called function(s)
  - Allow for highly-context sensitive flow-information



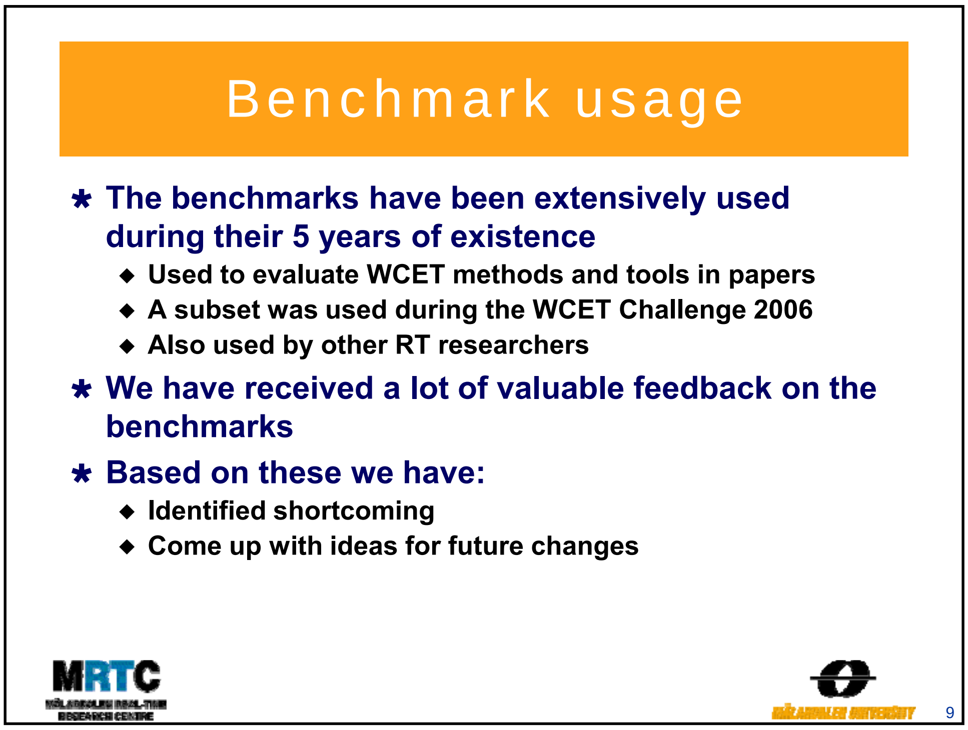
Example: snippet of scope-hierarchy graph for compress

Upper loop bounds

No. of iterations for main_initbuffer_L1 = 50
No. of iterations for main_compress_L1 = 8
No. of iterations for main_compress_L2 = 49
Max no. of iterations per invocation for main_compress_L2_L1 is 1
Max no. of iterations per invocation for main_compress_L2_cl_block_cl_hash_L1 is 16
Max no. of iterations per invocation for main_compress_L2_cl_block_cl_hash_L2 is 1

Example: compress .facit file

- Bounds valid for all possible inputs
  - Derived by exhaustive runs of all possible input value combinations
- Two levels of context-sensitivity
  - Global bounds - valid for each invocation of program
  - Local bounds - valid for each entry of loop in certain calling context (names refer to scopes in scope-hierarchy)
- Iteration bounds refer to loop headers
  - Some tools prefer bounds on loop bodies

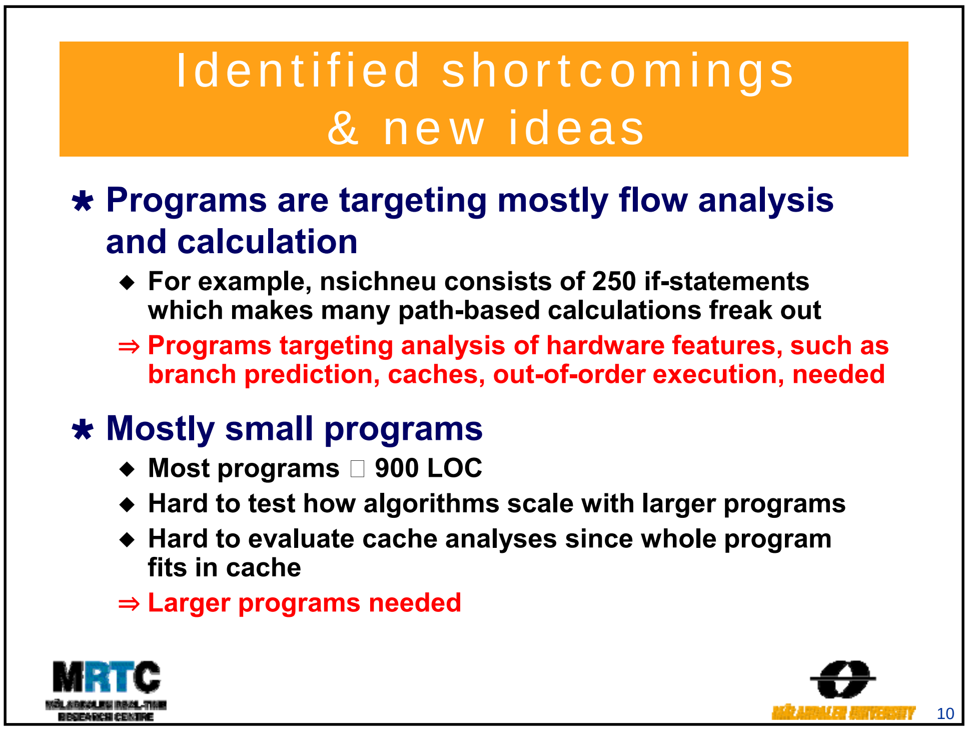


Benchmark usage

- The benchmarks have been extensively used during their 5 years of existence
  - Used to evaluate WCET methods and tools in papers
  - A subset was used during the WCET Challenge 2006
  - Also used by other RT researchers
- We have received a lot of valuable feedback on the benchmarks
- Based on these we have:
  - Identified shortcoming
  - Come up with ideas for future changes

MRTC MÄLARDALEN REAL-TIME RESEARCH CENTRE

MÄLARDALEN UNIVERSITY 9

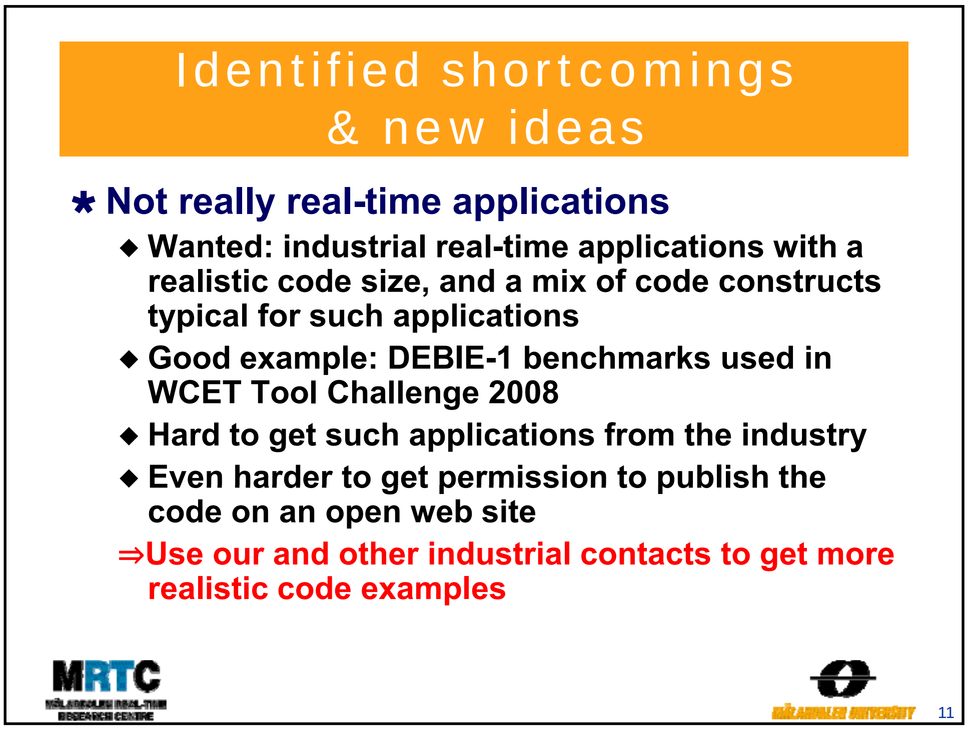


Identified shortcomings & new ideas

- Programs are targeting mostly flow analysis and calculation
  - For example, nsichneu consists of 250 if-statements which makes many path-based calculations freak out
  - ⇒ Programs targeting analysis of hardware features, such as branch prediction, caches, out-of-order execution, needed
- Mostly small programs
  - Most programs □ 900 LOC
  - Hard to test how algorithms scale with larger programs
  - Hard to evaluate cache analyses since whole program fits in cache
  - ⇒ Larger programs needed

MRTC MÄLARDALEN REAL-TIME RESEARCH CENTRE

MÄLARDALEN UNIVERSITY 10

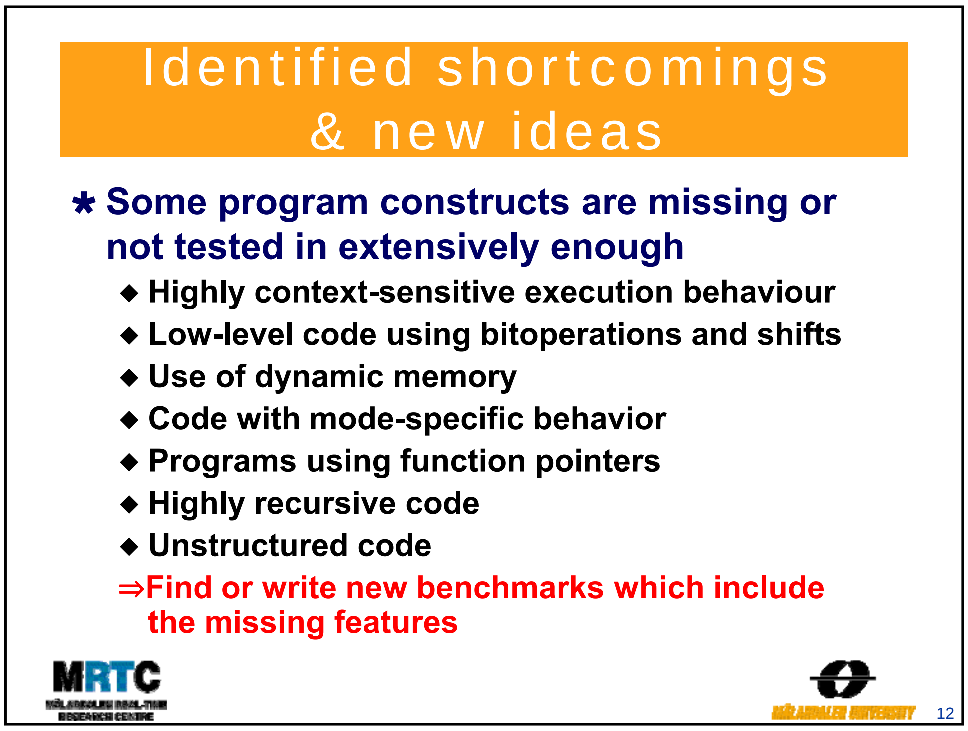


Identified shortcomings & new ideas

* Not really real-time applications
  - Wanted: industrial real-time applications with a realistic code size, and a mix of code constructs typical for such applications
  - Good example: DEBIE-1 benchmarks used in WCET Tool Challenge 2008
  - Hard to get such applications from the industry
  - Even harder to get permission to publish the code on an open web site

  ⇒Use our and other industrial contacts to get more realistic code examples

11

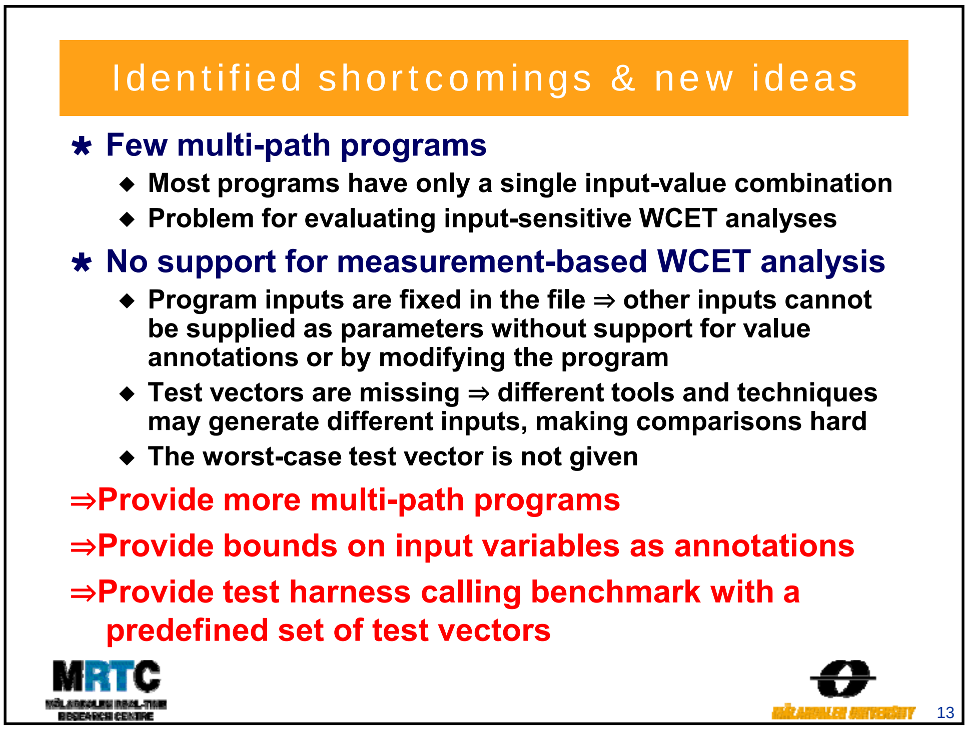


Identified shortcomings & new ideas

* Some program constructs are missing or not tested in extensively enough
  - Highly context-sensitive execution behaviour
  - Low-level code using bitoperations and shifts
  - Use of dynamic memory
  - Code with mode-specific behavior
  - Programs using function pointers
  - Highly recursive code
  - Unstructured code

  ⇒Find or write new benchmarks which include the missing features

12



Identified shortcomings & new ideas

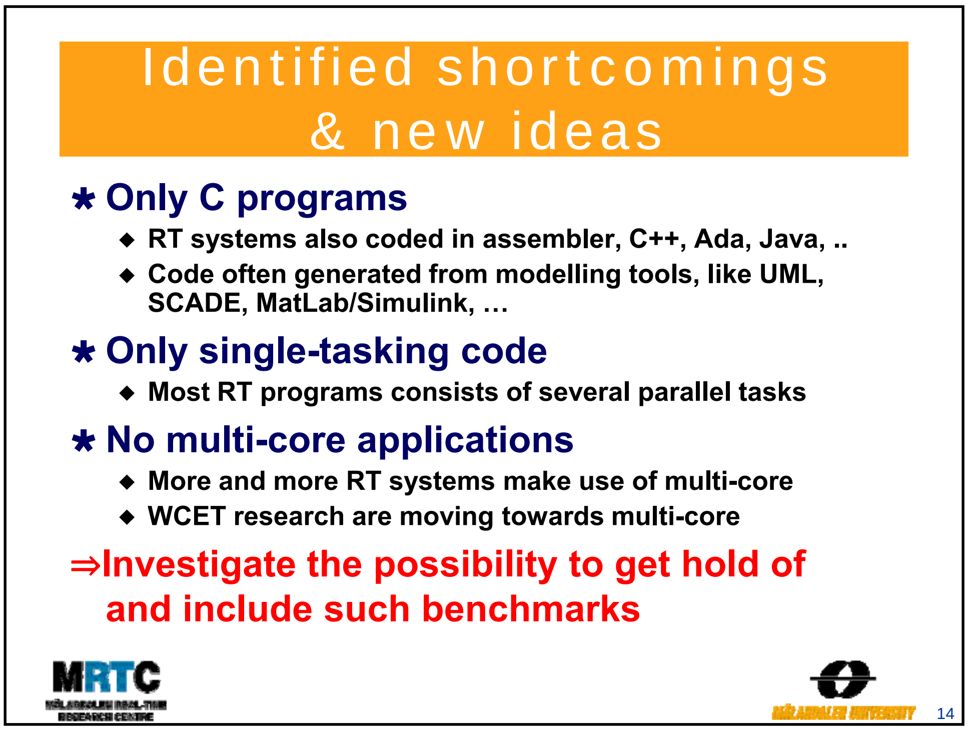
- **Few multi-path programs**
  - Most programs have only a single input-value combination
  - Problem for evaluating input-sensitive WCET analyses
- **No support for measurement-based WCET analysis**
  - Program inputs are fixed in the file ⇒ other inputs cannot be supplied as parameters without support for value annotations or by modifying the program
  - Test vectors are missing ⇒ different tools and techniques may generate different inputs, making comparisons hard
  - The worst-case test vector is not given

⇒Provide more multi-path programs

⇒Provide bounds on input variables as annotations

⇒Provide test harness calling benchmark with a predefined set of test vectors

13

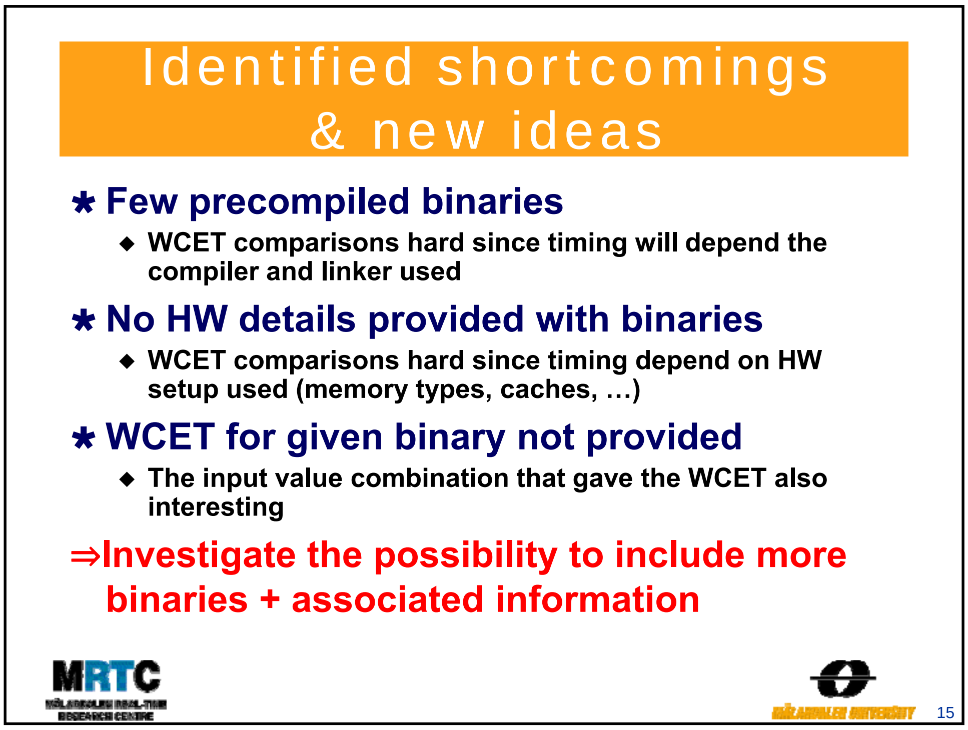


Identified shortcomings & new ideas

- **Only C programs**
  - RT systems also coded in assembler, C++, Ada, Java, ..
  - Code often generated from modelling tools, like UML, SCADE, MatLab/Simulink, …
- **Only single-tasking code**
  - Most RT programs consists of several parallel tasks
- **No multi-core applications**
  - More and more RT systems make use of multi-core
  - WCET research are moving towards multi-core

⇒Investigate the possibility to get hold of and include such benchmarks

14

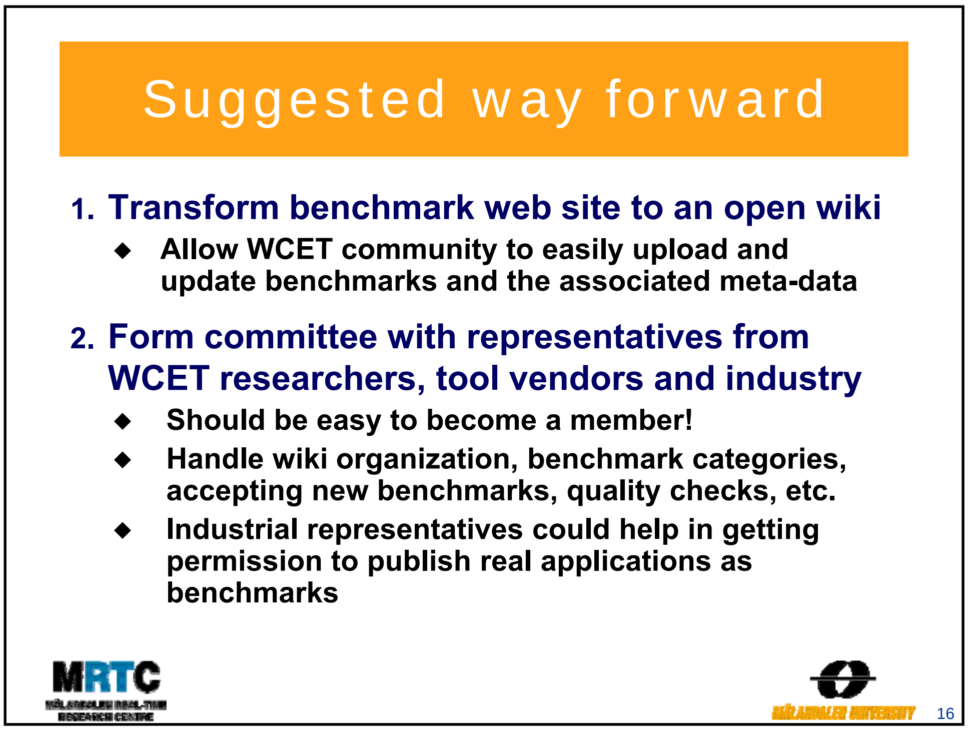


Identified shortcomings & new ideas

- **Few precompiled binaries**
  - WCET comparisons hard since timing will depend the compiler and linker used
- **No HW details provided with binaries**
  - WCET comparisons hard since timing depend on HW setup used (memory types, caches, …)
- **WCET for given binary not provided**
  - The input value combination that gave the WCET also interesting

⇒Investigate the possibility to include more binaries + associated information

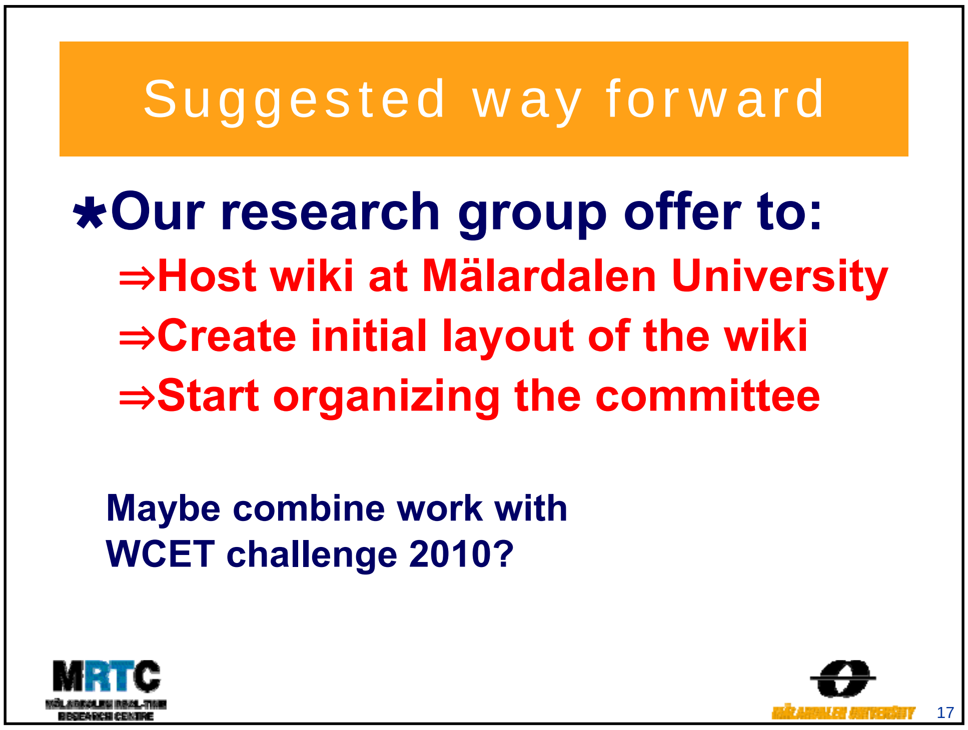
15



Suggested way forward

1. **Transform benchmark web site to an open wiki**
   - Allow WCET community to easily upload and update benchmarks and the associated meta-data
2. **Form committee with representatives from WCET researchers, tool vendors and industry**
   - Should be easy to become a member!
   - Handle wiki organization, benchmark categories, accepting new benchmarks, quality checks, etc.
   - Industrial representatives could help in getting permission to publish real applications as benchmarks

16



Suggested way forward

- ✶Our research group offer to:
  - ⇒Host wiki at Mälardalen University
  - ⇒Create initial layout of the wiki
  - ⇒Start organizing the committee

Maybe combine work with
WCET challenge 2010?

MRTC MÄLARDALEN REAL-TIME RESEARCH CENTRE

MÄLARDALEN UNIVERSITY 17

Thank you for your attention!

Questions or comment?