http://www.mrtc.mdh.se/projects/WCC/2011/

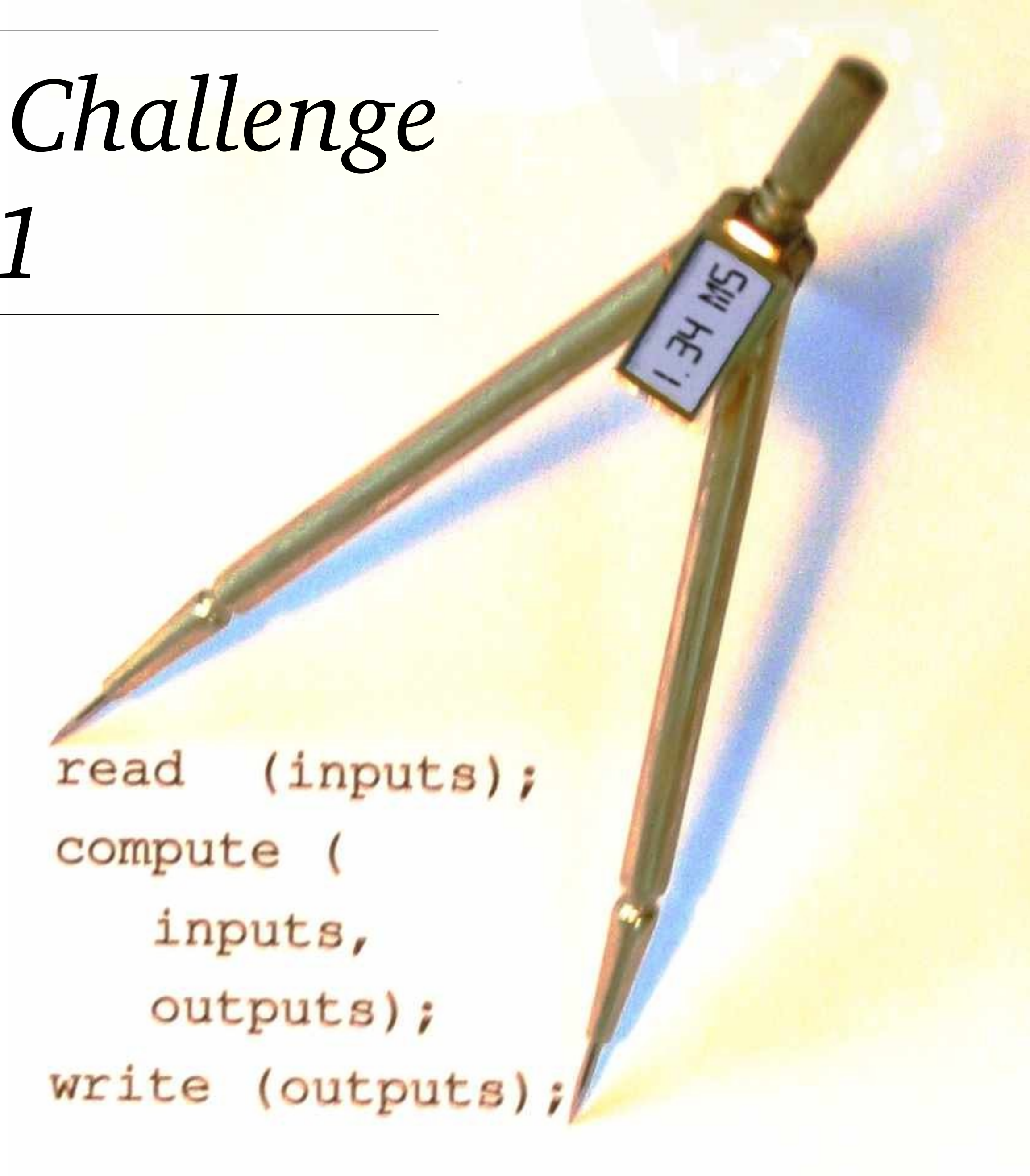
3rd Worst Case Challenge WCC'11

aims and participants

tools and benchmarks

problems and results

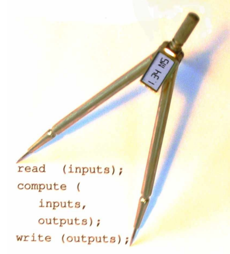
experiences and conclusions



Presented by
Björn Lisper, Mälardalen University
Niklas Holsti, Tidorum Ltd

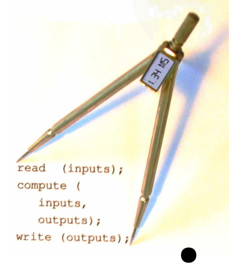
Authors

Reinhard von Hanxleden, Niklas Holsti, Björn Lisper,
Erhard Ploedereder, Reinhard Wilhelm (Eds.),
Armelle Bonenfant, Hugues Cassé, Sven Bünte,
Wolfgang Fellger, Sebastian Gepperth, Jan Gustafsson,
Benedikt Huber, Nazrul Mohammad Islam,
Daniel Kästner, Raimund Kirner, Laura Kovács,
Felix Krause, Marianne de Michiel, Mads Christian Olesen,
Adrian Prantl, Wolfgang Puffitsch, Christine Rochange,
Martin Schoeberl, Simon Wegener, Michael Zolda,
Jakob Zwirchmayr



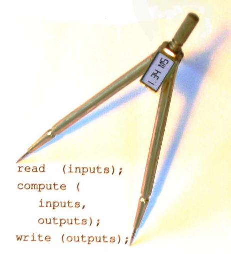
WCET Challenge: Aims

- Use many WCET tools on the same benchmark programs
  - challenge tools with new benchmarks
  - enable cross-tool comparisons along several dimensions
  - demonstrate novel abilities of tools
  - present wide spectrum of tools to potential users
- Comparison dimensions include:
  - automatic analysis vs. manually written annotations
    - eg. loop bounds and infeasible paths
  - expressiveness and usability of annotation mechanism
    - eg. using source-code names vs. machine addresses
  - precision of WCET bounds or estimates
    - difficulty: true WCETs known only for small benchmarks
- Both static analysis tools and measurement-based tools
  - difficulty: few benchmarks have test suites

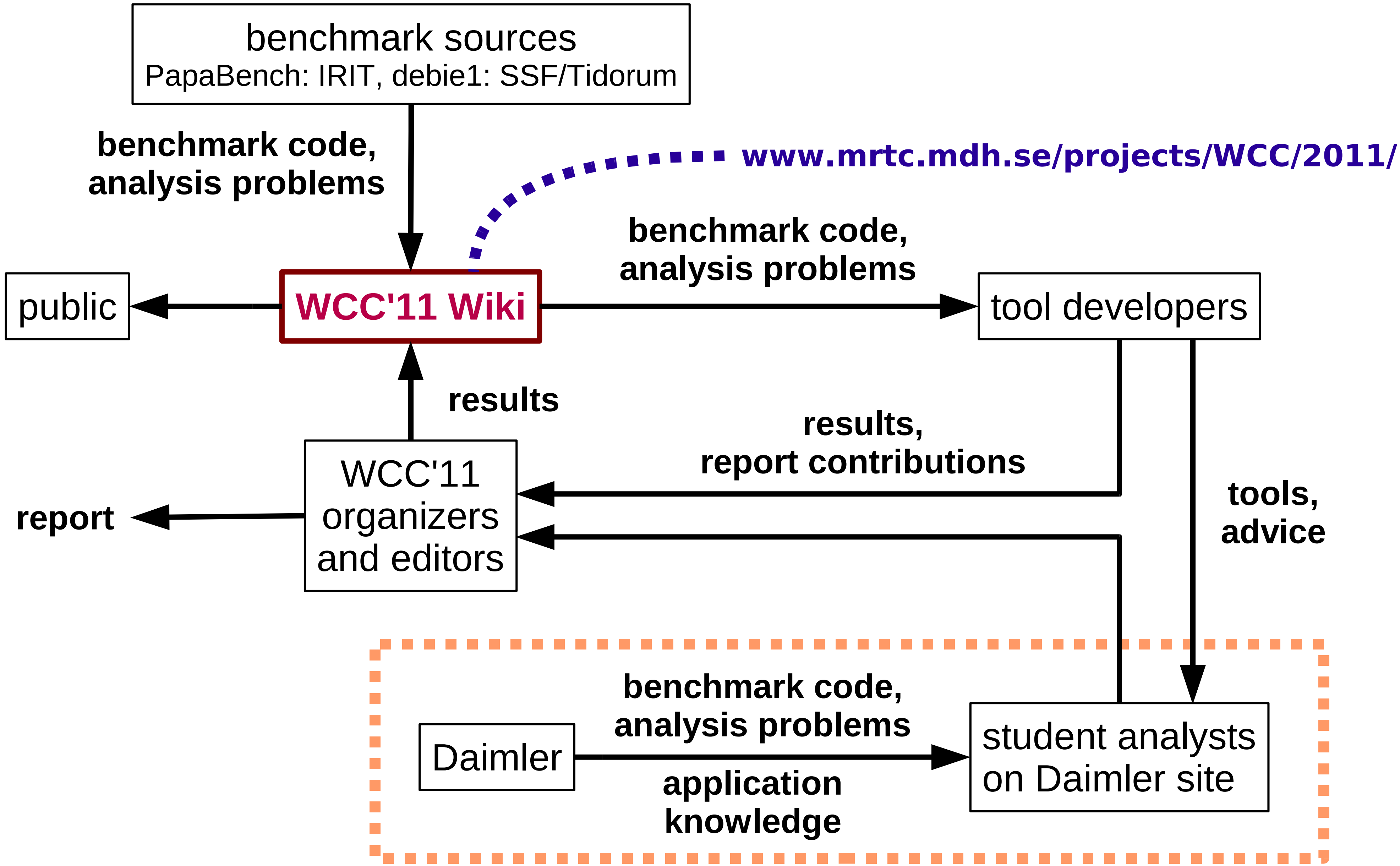


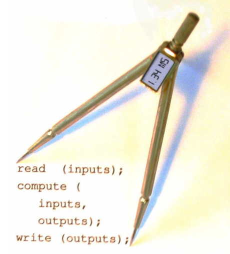
From 2006 through 2008 to 2011

- First Challenge 2006 – pioneering success
  - several WCET tools, benchmarks, processors
  - few comparable combinations of benchmark + processor
- Second Challenge 2008
  - one common target processor (system)
  - better definition of benchmarks, problems, and results
  - include pure flow-analysis problems and new benchmarks
- Third Challenge 2011
  - one simple and one complex common target processor
  - other processors allowed, eg. Java processor
  - benchmarks: some old, some new, some borrowed
  - experiment at Daimler automotive:
    - students use WCET tools on industrial code
    - gauge learning effort, usability, skills needed
  - see Wiki at www.mrtc.mdh.se/projects/WCC/2011/



WCC'11 process

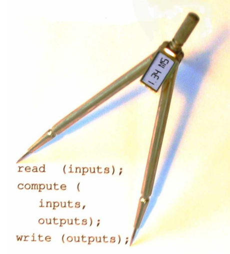




Participants through the ages

Tool	*Source*	2006	2008	2011
aiT	AbsInt GmbH	SWb		SWb
Astrée	AbsInt GmbH			SFs
Bound-T	Tidorum Ltd	SWb	SWb	SWb
Chronos	National Univ. Singapore	SWsb		
FORTAS	TU Vienna			Hsb
METAMOC	Aalborg University			(SWsb)
MTime	TU Vienna	(H)	H	
oRange + OTAWA	IRIT		SWsb	SWsb
RapiTime	Rapita Systems Ltd		(Hsb)	
SWEET	Mälardalen University	SWb		SFs
TimeWeaver	AbsInt GmbH			Hb
TuBound	TU Vienna		SWsb	SWsb
WCA	TU Vienna, TU Denmark			SWc
wcc	Dortmund U Technology		SFsb	

S--	Static	-W-	WCET & flow	--s	Source	--c	Bytecode
H--	Hybrid	-F-	Flow only	--b	Binary		
		(---)	No results	--sb	Source (flow) and binary (WCET)		



Target processors

the "simple" common processor

Processor	*Tools*	2006	2008	2011
ARM7	aiT, Bound-T, METAMOC, OTAWA, RapiTime	**1**	**4**	**4**
ARM9	SWEET	**1**		
C16x	aiT,TuBound	**1**	**1**	**1**
ERC32 (SPARC)	Bound-T	**1**		
JOP (Java)	WCA			**1**
MPC5553/5554	aiT, OTAWA			**2**
MPC565	aiT	**1**		
Renesas H8/300	Bound-T	**1**		
SimpleScalar	Chronos	**1**		
TriCore 1796	FORTAS			**1**

the "complex" common processor(s)

Number of tools: **1** N = 1 **2** N = 2 **N** N = 3 or more

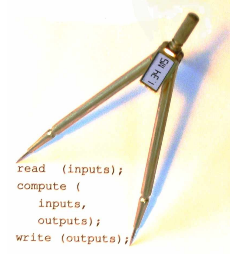
Benchmarks

Benchmark	*Source (immediate)*	2006	2008	2011
Mälardalen bm's	Mälardalen University	**Os**		
PapaBench	TRACES/IRIT	**Os**		**Os**
rathijit bm's	Saarland University		**Os**	
debie1	SSF/Tidorum Ltd		**Rs**	**Rs**
Daimler bm	Daimler			**Ps**

Os	Open source
Rs	Restricted, source provided
Ps	Proprietary, source shown to analysts

For WCC'2011, all benchmarks are “real” programs:

- PapaBench: from Paparazzi Unmanned Aerial Vehicle controller
- debie1: from DEBIE-1 satellite-based scientific instrument
- Daimler: from truck control system



Benchmark details

- PapaBench
  - two programs (autopilot, fly-by-wire), originally 2 CPUs
  - C; 5020 lines, 1521 semicolons (“airborne” part)
  - single thread per program, cyclic scheduler + interrupts
- debie1
  - one program, originally six threads under preemptive RTK
  - C; 10228 lines, 1748 semicolons (excluding test harness)
  - test suite included (no I/O or RTK needed)
- Daimler
  - control system for trucks, eg. collision detection
  - C; size not revealed
  - parts analysed: an interrupt handler; an initialization routine; two calculation routines; one complete task

“Analysis Problem” definitions

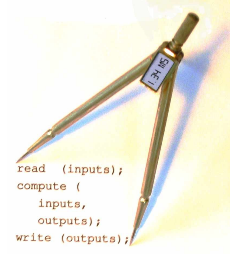
- Conditions + question
- Flow analysis problems
  - how many times can X occur under these conditions?
  - eg. how many times can `Foo` call `Bar` ?
- WCET analysis problems
  - give a WCET bound on Foo under these conditions
- “Subprogram”
  - the root procedure/function (or part of it, in some cases)
- “Inputs”
  - bounds on values of relevant input data
- “Other constraints”
  - other conditions on the scenarios/executions to be included
  - for example: no call of `Reboot` is executed

Example analysis problems

- debie1, problem 2a/F1 (flow analysis):
  - assuming that:

    `telemetry_pointer < telemetry_end_pointer` and
    `telemetry_pointer != &telemetry_data.time`
  - how many times can `TM_InterruptService` call `Send_ISR_Mail` ?
- PapaBench AutoPilot, problem A2a (WCET analysis):
  - assuming that:

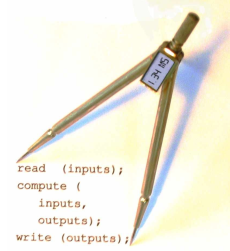
    `pprz_mode = PPRZ_MODE_HOME` (value 3)
  - what is a WCET bound for `navigation_update` ?
- Daimler benchmark:
  - analysis problems not shown on the Wiki
  - results give WCET of four functions



Questions and answers, overall

Benchmark	debie1		PapaBench		Daimler
Type of question	Flow	WCET	Flow	WCET	WCET
Number of questions	15	22	6	11	4
aiT	15	22	3	11	4
Astrée	15				
Bound-T	14	18	6	11	
FORTAS				5	
METAMOC					
OTAWA	8	15	5	11	4
SWEET			6		
TimeWeaver		6			
TuBound	15	18	1	10	
WCA		13		11	

Table 5: Number of posed and answered analysis problems in WCC'11.

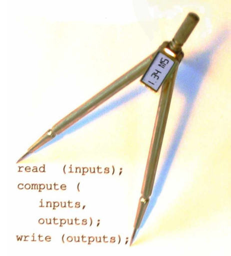


Obstacles in the benchmarks

- Floating point computations (debie1, PapaBench)
  - loops in SW FP libraries difficult to bound automatically
    - annotations needed
  - few tools do value analysis of FP variables
    - loops with FP conditions must be annotated
- Source code missing for library functions (FP & others?)
  - source-code analyzers (Astrée, oRange) need it
  - manual annotation easier if source code is available
- MPC (PowerPC) instruction set subsets
  - debie1 code contains Variable Length Encoding instructions
    - VLE was not supported yet in OTAWA
- Infinite loops (eg. Daimler benchmark “task” code)
- Large number of tests in debie1, giving large traces
  - hampered TimeWeaver

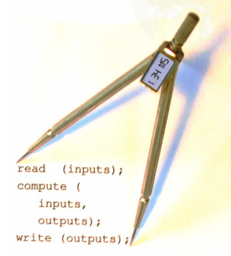
Obstacles in the tools

- MPC instruction set is complex, has subsets
  - debie1 code contains Variable Length Encoding instructions
    - VLE was not supported yet in OTAWA
- TuBound failed to install at Daimler
  - depends on other SW, some with licensing restrictions
- SWEET did not attempt Daimler benchmark
  - C-to-ALF translator hard to install, many dependencies
- TimeWeaver had problems with tracing hardware
- New or prototype tools, still in development:
  - FORTAS
  - METAMOC



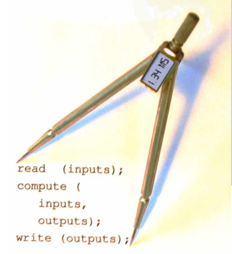
Tool developments during WCC'11

- aiT: increase automation, reduce annotations
- Astrée: streamlined export of flow information
- METAMOC: “gained many improvements”
- OTAWA: began extension to full MPC instruction set
- SWEET: new treatment of data at absolute address
- Tidorum (Bound-T): new tools for ET measurement
- TimeWeaver: handle incomplete traces, multi-exit funcs
- WCA: improved annotation language



Benchmark ports during WCC'11

- Done in order to enable participation in WCC'11
- PapaBench
  - port to TriCore 1796 (FORTAS team)
  - port to C167 (TuBound team)
  - translation into ALF (SWEET team)
  - Java version improved for JOP (WCA team)
- debie1
  - port to C167 (TuBound team)
  - port to MPC5554 (Simon Wegener, AbsInt)
  - translation to Java / JOP (WCA team)
- Shows significant effort and interest of participants



Results: open benchmarks

- Can compare results of three tools on the same code:
  - tools: aiT, Bound-T, OTAWA
  - benchmarks: debie1, PapaBench
  - processor: ARM7 with single-cycle memory access
- Some differences are seen, reasons not always clear...
  - differences in flow analysis? eg. infeasible paths
  - different interpretations of Analysis Problem conditions?
  - differences in processor modelling?
- Comparison to debie1 measured ET from Tidorum
  - measurements came very late (not originally planned)
  - **NOT** worst-case measurements, in general
  - still, comparison to aiT & Bound-T results revealed errors:
    - in analysis annotations (aiT and Bound-T)
    - in measurement procedures (debie1 harness code)

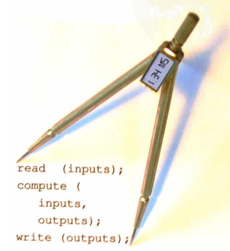
Example of results for ARM7

Note: “measured” is generally **not** the worst case!

BM problem	aiT	Bound-T	OTAWA	measured
debie1:				
1-T1	342	333	332	303
2a-T1	100	93	139	93
3a-T1	2 664	2 692	4 101	2 340
4b-T1	215	214	210	206
5b-T1	38 798	39 825	55 883	37 463
6c-T1	40 143	42 285	-	19 726
PapaBench:				
A1	1 716	1 660	1 358	
A2b	31 482	37 181	38 112	
A6	12 051	17 378	17 422	

The Daimler Experiment

- Analysts: Stuttgart students (Krause, Geppert, Fellger)
  - no earlier experience of WCET analysis or this benchmark
  - access to Daimler experts on the benchmark code
  - remote support from tool providers
- Tools: aiT, OTAWA (oRange not used)
  - TuBound installation problems at Daimler
  - SWEET feared same, did not try
- Target: MPC5553 (PowerPC; no external memory)
- Benchmark: one program, part of truck control system
- Analysis Problems (all WCET questions):
  - interrupt handler (INTERR), no loops, one call
  - initialization routine (INIT), no loops, no calls
  - calculation routines (CALC1, 2), some loops and calls
  - complete task (TASK), endless loop: call subtasks, suspend



Results: Daimler Experiment

- Students report on usability
  - aiT: pretty straightforward to use; no major problems; loop-bound over-estimation reasonable
  - OTAWA: could not handle infinite loops; sometimes crashed; needs oRange for loop-bounds analysis (not used here)
- Flow analysis
  - context-dependent loop-bounds important
  - available in aiT, not in OTAWA
- WCET comparison aiT - OTAWA
  - actual ETs not measured
  - the target is MPC5553, but OTAWA supports MPC5554
    - aiT analysis was executed for both models
    - OTAWA result usually < aiT result, but...
  - OTAWA results (for other code) sometimes < measured ET
    - OTAWA's model of MPC554 needs correction?

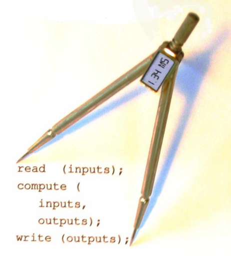
Daimler Experiment: WCETs

Entry point	aiT		OTAWA
	MPC5553	MPC5554	MPC5554
INTERR	524	204	113
INIT	1 055	494	218
CALC1	2 124	830	722
CALC2	16 507	6 218	7 991
TASK			

WCET for “TASK” not reported because this entry point has an infinite loop.

Summary of WCC'11

- **Successes** (goals set after WCC'08):
  - more participants, in particular many from WCC'06 again
  - some comparable results:
    - same processor, same benchmark, different tools
  - some continuity in benchmarks and targets
  - but also new participants, targets, benchmarks
  - evaluation of usability with novice users (Daimler experim.)
- **Failures**:
  - not many comparable results
  - no validated WCETs for reference
    - measured ETs for only one benchmark
  - few participants for the “complex processor” (MPC)
  - few participants in the Daimler experiment
    - non-disclosure requirements, hinder eg. porting
  - Worst Case Challenge is still sporadic, not continuous



Reactions from WCC'11 participants

- “Our research benefits from the extended pool of benchmarks” (FORTAS)
- “[now] much clearer [how] to improve the tool” (METAMOC)
- “a good source of inspiration” (TimeWeaver)
- “we got many new ideas for improving the tool” (WCA)

Suggestions for future Challenges

- Split into two phases?
  1. Flow analysis phase, giving loop bounds etc.
     - combine to give best (tightest) flow constraints
     - eg. only SWEET could find bounds on floating-point loops
  2. WCET analysis phase
     - all tools use the same (best) flow information
     - separate flow analysis from processor modelling
- Several participants supported the two-phase idea
- Usability evaluation (“Daimler Experiment”)
  – should be at start of Challenge
    - to give tool providers time to respond
  – should preferably use open benchmark
    - so that experiences can be discussed openly
- Better support for measurement-based tools: test cases!

Thanks

Thanks to all Challenge participants!