SAT Solving for Model Checking and Beyond

Armin Biere
Johannes Kepler University
Linz, Austria

HVC'15

11^{th} Haifa Verification Conference

IBM Research, Haifa, Israel

Thursday, 19th November, 2015

JOHANNES KEPLER
UNIVERSITY LINZ

HVC2015

November 17 – 19 Haifa, Israel

Tutorials: November 16

Home | Program | Tutorials | HVC Award | Social Events | Venue & Travel | Organization

HVC Award

The HVC award is given to the most influential work in the last five years in formal verification, simulation, and testing. The award is not limited to influential articles; it can also be a system or a collection of activities that promote the area.

The HVC award committee has decided to give the award this year to **Prof. Armin Biere** from Johannes Kepler University in Austria.

Since Biere's PhD graduation in 1997, he made pivotal contributions in formal verification, one of which --- bounded model checking --- was recently selected as the most influential article in the 20 years of TACAS. This technique is still being used in numerous EDA companies, and also led to similar ideas in verification of software. In addition, he is the developer of numerous award-winning SAT, bitvector, arrays and QBF solvers. Such solvers developed by him or under his guidance rank at the top of many international competitions and were awarded 42 medals including 24 gold medals.

Biere is one of the editors of the 980-pages "Handbook of Satisfiability", the chair of the SAT association, the founder and organizer of the Hardware Model Checking Competition (HWMCC), the inventor of the now ubiquitous AIG format for model-checking, and also has served in the last two years as an informal advisor of D. Knuth for SAT-related issues.

By awarding Prof. Biere, the committee recognizes his major contributions to the formal verification and computational logic communities.

Call for papers

Submissions

Registration

Keynote Speakers

- **Patrice Godefroid, Microsoft Research**
- **Stephen Bailey, Director of Emerging Technologies, Mentor Graphics**
- **Prof. Mooly Sagiv, Tel Aviv University**
- **Bodo Hoppe, Hardware Verification, IBM**

Armin Biere. μcke - efficient μ-calculus model checking. In Orna Grumberg, editor, Computer Aided Verification, 9th International Conference, CAV'97, Haifa, Israel, June 22-25, 1997. Volume 1254 of Lecture Notes in Computer Science., Springer (1997) 468–471

μcke – Efficient μ-Calculus Model Checking

Armin Biere[1]

armin@ira.uka.de, Institut für Logik, Komplexität und Deduktionssysteme,
Universität Karlsruhe, Am Fasanengarten 5, D-76128 Karlsruhe, Germany

Abstract. In this paper we present an overview of the verification tool μcke. It is an implementation of a BDD-based μ-calculus model checker and uses several optimization techniques that are lifted from special purpose model checkers to the μ-calculus. This gives the user more expressibility without loosing efficiency.

Introduction

In [5] μ-calculus model checking with BDDs has been proposed as a general framework for various verification problems like model checking of LTL and CTL or testing for bisimulation equivalence and language containment. With a μ-calculus model checker all these verification tasks could be handled with one tool. Also some applications of

Arne Borälv. The Industrial Success of Verification Tools Based on Stålmarck's Method. In Orna Grumberg, editor, Computer Aided Verification, 9th International Conference, CAV'97, Haifa, Israel, June 22-25, 1997. Volume 1254 of Lecture Notes in Computer Science., Springer (1997) 468–471

The Industrial Success of Verification Tools Based on Stålmarck's Method

Arne Borälv
arne@lk.se

Logikkonsult NP AB,
Swedenborgsgatan 2,
S-118 48 Stockholm, Sweden,
http://www.lk.se

Abstract. Stålmarck's Method is a patented natural deduction proof method with a novel proof-theoretic notion of *proof depth*, defined as the largest number of nested assumptions in the proof. An implementation of the method, called Prover, has been used as proof engine in various commercial tools since 1990, and is now integrated in a formal verification framework called NP-Tools. Prover searches for shallow sub-formula proofs, which has proven to be an efficient strategy for solving many industrial problems, the largest of which today consists of several 100,000's of sub-formulas. Stålmarck's method is in industrial use, for instance in the areas of telecom service specification analysis, analysis of

Symbolic Model Checking without BDDs*

Armin Biere[1], Alessandro Cimatti[2], Edmund Clarke[1], and Yunshan Zhu[1]

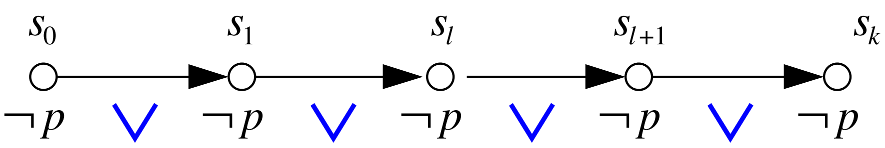
[1] Computer Science Department, Carnegie Mellon University
5000 Forbes Avenue, Pittsburgh, PA 15213, U.S.A
{Armin.Biere,Edmund.Clarke,Yunshan.Zhu}@cs.cmu.edu
[2] Istituto per la Ricerca Scientifica e Tecnologica (IRST)
via Sommarive 18, 38055 Povo (TN), Italy
cimatti@irst.itc.it

Abstract. Symbolic Model Checking [3, 14] has proven to be a powerful technique for the verification of reactive systems. BDDs [2] have traditionally been used as a symbolic representation of the system. In this paper we show how boolean decision procedures, like Stålmarck's Method [16] or the Davis & Putnam Procedure [7], can replace BDDs. This new technique avoids the space blow up of BDDs, generates counterexamples much faster, and sometimes speeds up the verification. In addition, it produces counterexamples of minimal length. We introduce a *bounded model checking* procedure for LTL which reduces model checking to propositional satisfiability. We show that bounded LTL model checking can be done without a tableau construction. We have implemented a model checker **BMC**, based on bounded model checking, and preliminary results are presented.

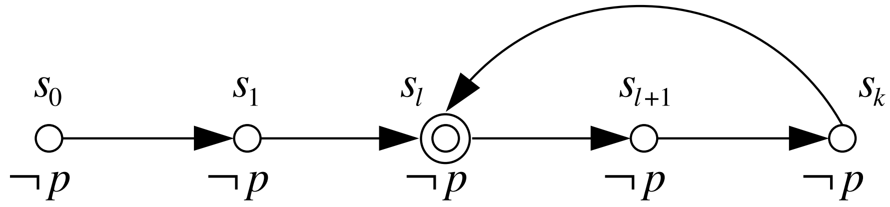
Bounded Model Checking

[BiereCimattiClarkeZhu-TACAS'99]

- look only for counter example made of k states   "k" = bound



or



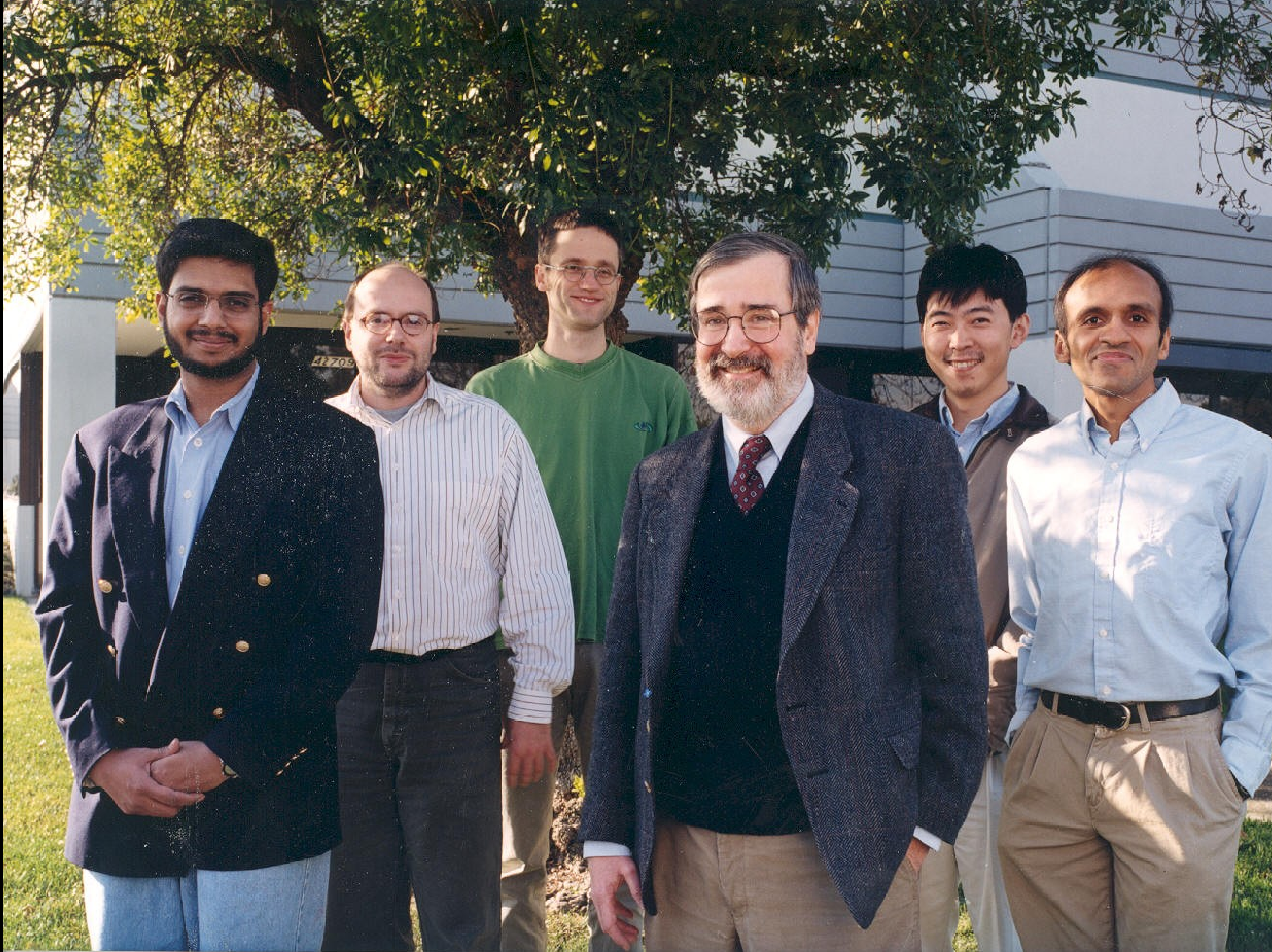
- simple for <u>safety properties</u> p invariantly true

$$I(s_0) \wedge T(s_0, s_1)) \wedge \cdots \wedge T(s_{k-1}, s_k) \wedge \bigvee_{i=0}^{k} \neg p(s_i)$$

- harder for <u>liveness properties</u> p eventually true

$$I(s_0) \wedge T(s_0, s_1)) \wedge \cdots \wedge T(s_{k-1}, s_k) \wedge \bigwedge_{i=0}^{k} \neg p(s_i) \wedge \bigvee_{l=0}^{k} T(s_k, s_l)$$

- compute and bound k by diameter


42709

Google

Scholar

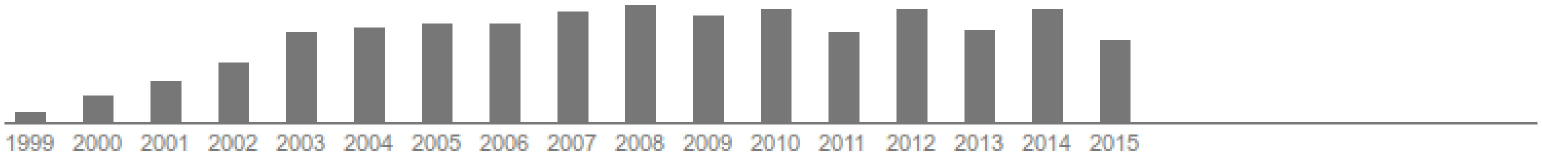
Edit

Export

Symbolic model checking without BDDs

[PDF] from cmu.edu

Authors	Armin Biere, Alessandro Cimatti, Edmund Clarke, Yunshan Zhu
Publication date	1999/1/1
Book	Tools and Algorithms for the Construction and Analysis of Systems
Pages	193-207
Publisher	Springer Berlin Heidelberg
Description	Abstract Symbolic Model Checking [3],[14] has proven to be a powerful technique for the verification of reactive systems. BDDs [2] have traditionally been used as a symbolic representation of the system. In this paper we show how boolean decision procedures, like Stålmarck's Method [16] or the Davis & Putnam Procedure [7], can replace BDDs. This new technique avoids the space blow up of BDDs, generates counterexamples much faster, and sometimes speeds up the verification. In addition, it produces counterexamples of minimal ...
Total citations	Cited by 2076



Scholar articles

Symbolic model checking without BDDs

A Biere, A Cimatti, E Clarke, Y Zhu - Tools and Algorithms for the Construction and Analysis ..., 1999

Cited by 2076 - Related articles - All 38 versions

Replacing Testing with Formal Verification in Intel® Core™ i7 Processor Execution Engine Validation

Roope Kaivola, Rajnish Ghughal, Naren Narasimhan, Amber Telfer, Jesse Whittemore, Sudhindra Pandav, Anna Slobodová, Christopher Taylor, Vladimir Frolov, Erik Reeber, and Armaghan Naik

Intel Corporation, JF4-451, 2111 NE 25th Avenue, Hillsboro, OR 97124, USA

Abstract. Formal verification of arithmetic datapaths has been part of the established methodology for most Intel processor designs over the last years, usually in the role of supplementing more traditional coverage oriented testing activities. For the recent Intel® Core™ i7 design we took a step further and used formal verification as the primary validation vehicle for the core execution cluster, the component responsible for the functional behaviour of all microinstructions. We applied symbolic simulation based formal verification techniques for full datapath, control and state validation for the cluster, and dropped coverage driven testing entirely. The project, involving some twenty person years of verification work, is one of the most ambitious formal verification efforts in the hardware industry to date. Our experiences show that under the right circumstances, full formal verification of a design component is a feasible, industrially viable and competitive validation approach.

1 Introduction

Copyrighted Material

A. Bouajjani and O. Maler (Eds.): CAV 2009, LNCS 5643, pp. 414–429, 2009.
© Springer-Verlag Berlin Heidelberg 2009

6 Formal Verification Value Proposition

The conventional wisdom about formal verification in industrial context is easy to spell

a few, in both cases leading to a perceived low return on investment. The areas where projects have routinely chosen to do formal verification have then been limited to those where an uncaught problem would be so visible and costly that the extra effort of doing formal verification can be justified. As a positive exception, SAT-based bounded model checking has been very successfully used as a bug-hunting tool in targeted areas.

The third usage model, mixing formal and dynamic techniques on validating a sin-

Impact of BMC

- widespread use in industry (EDA)
  - industry embraced bounding part immediately
  - original *industrial* reservations: using SAT vs ATPG
  - original *academic* reservations: incompleteness?
- BMC relies on efficient SAT (SMT) solving
  - breakthroughs in SAT: CDCL '96, VSIDS '01, ...
  - encouraged investment in SAT / SMT research
- extensions to *non-boolean* domains and SW
  - bounding reduces complexity / decidability
- extensions to *completeness*
  - diameter checking, *k*-induction, interpolation
  - SAT based model checking *without* unrolling: IC3

JOHANNES KEPLER
UNIVERSITÄT LINZ

A Short Story on 15 years of

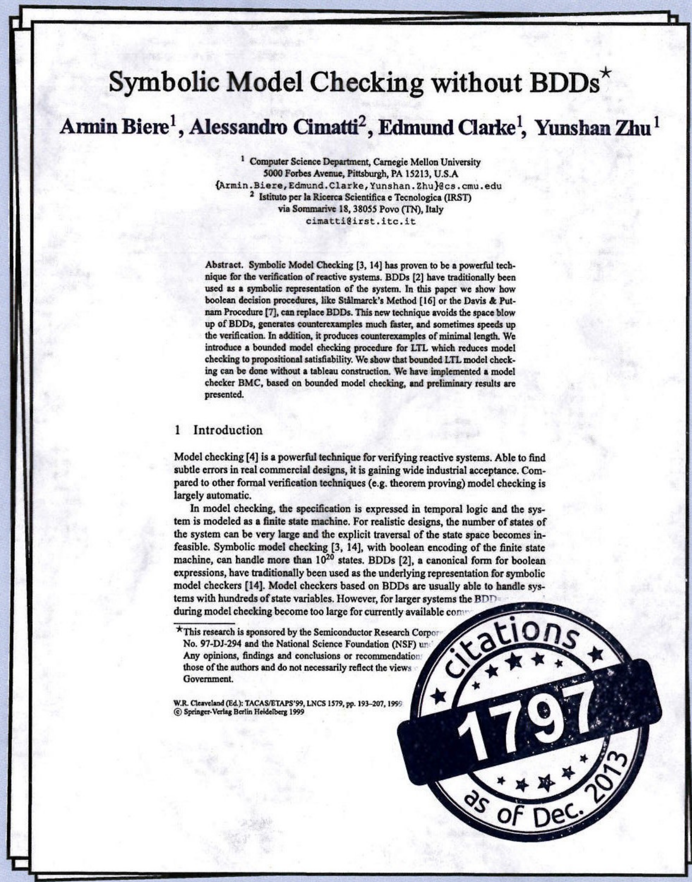
Bounded Model Checking

- 1997: interest and capacity of BDDs stalled
  but there were success stories of other techniques
- *Ed Clarke* hired *Yunshan Zhu* & *Armin Biere* as Post-Docs:
  Use SAT for Symbolic Model Checking!
- struggled for 10 months to come up with something that could replace / improve BDDs (mainly looked at QBF then)
- *Alessandro Cimatti* came to an AI conference in Pittsburgh and at lunch (at an Indian Restaurant) we realized, that in AI Planing they **do not care about completeness**
  What if we apply this to model checking?
  How to handle temporal logic?
- After one afternoon for the theory and 3 months of implementation and benchmarking later: *TACAS submission*

JOHANNES KEPLER
UNIVERSITÄT LINZ

AWARD

Most influential paper
in the first 20 years of TACAS



Symbolic Model Checking without BDDs*

Armin Biere[1], Alessandro Cimatti[2], Edmund Clarke[1], Yunshan Zhu[1]

[1] Computer Science Department, Carnegie Mellon University
5000 Forbes Avenue, Pittsburgh, PA 15213, U.S.A
{Armin.Biere,Edmund.Clarke,Yunshan.Zhu}@cs.cmu.edu
[2] Istituto per la Ricerca Scientifica e Tecnologica (IRST)
via Sommarive 18, 38055 Povo (TN), Italy
cimatti@irst.itc.it

Abstract. Symbolic Model Checking [3, 14] has proven to be a powerful technique for the verification of reactive systems. BDDs [2] have traditionally been used as a symbolic representation of the system. In this paper we show how boolean decision procedures, like Stålmarck's Method [16] or the Davis & Putnam Procedure [7], can replace BDDs. This new technique avoids the space blow up of BDDs, generates counterexamples much faster, and sometimes speeds up the verification. In addition, it produces counterexamples of minimal length. We introduce a bounded model checking procedure for LTL which reduces model checking to propositional satisfiability. We show that bounded LTL model checking can be done without a tableau construction. We have implemented a model checker BMC, based on bounded model checking, and preliminary results are presented.

1 Introduction

Model checking [4] is a powerful technique for verifying reactive systems. Able to find subtle errors in real commercial designs, it is gaining wide industrial acceptance. Compared to other formal verification techniques (e.g. theorem proving) model checking is largely automatic.

In model checking, the specification is expressed in temporal logic and the system is modeled as a finite state machine. For realistic designs, the number of states of the system can be very large and the explicit traversal of the state space becomes infeasible. Symbolic model checking [3, 14], with boolean encoding of the finite state machine, can handle more than 10^{20} states. BDDs [2], a canonical form for boolean expressions, have traditionally been used as the underlying representation for symbolic model checkers [14]. Model checkers based on BDDs are usually able to handle systems with hundreds of state variables. However, for larger systems the BDD
during model checking become too large for currently available com

*This research is sponsored by the Semiconductor Research Corpor
No. 97-DJ-294 and the National Science Foundation (NSF) un
Any opinions, findings and conclusions or recommendation
those of the authors and do not necessarily reflect the views
Government.

W.R. Cleaveland (Ed.): TACAS/ETAPS'99, LNCS 1579, pp. 193–207, 1999

© Springer-Verlag Berlin Heidelberg 1999

April 8th 2014, Grenoble

W. R. Cleaveland II   Lenore Zuck   Kim Larsen

The Steering Committee of TACAS

SAT Based Model Checking

- BMC
- *k*-induction
- Abstractions / CEGAR
- Interpolation
- IC3

Abstract Modern satisfiability (SAT) solvers have become the enabling technology of many Model Checkers. In this chapter, we will focus on those techniques most relevant to industrial practice. In *Bounded Model Checking* (BMC), a transition system and a property are jointly unwound for a given number k of steps to obtain a formula that is satisfiable if there is a counterexample for the property up to length k. The formula is then passed to an efficient SAT solver. The strength of BMC is *refutation*: BMC has been used to discover subtle flaws in digital systems. We cover the application of BMC to both hardware and software systems, and to hardware/software co-verification. We also discuss means to make BMC complete, including k-induction, Craig interpolation, abstraction refinement techniques and inductive techniques with iterative strengthening.

SAT Based Model Checking

Armin Biere, Daniel Kröning

Handbook of Model Checking

Edmund Clarke, Thomas Henzinger, Helmut Veith, *editors*

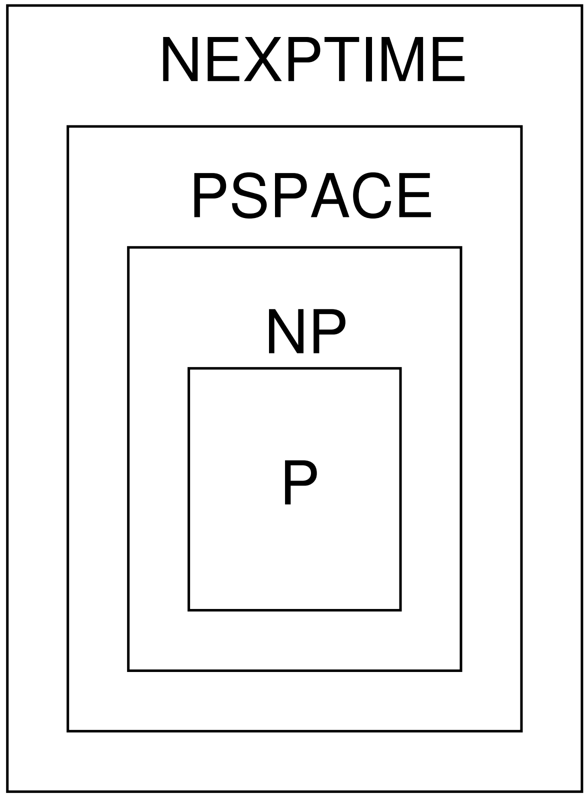
JKU JOHANNES KEPLER UNIVERSITÄT LINZ

Lessons from BMC

- simple but useful ideas are *very* controversial
  - hard to get accepted (literally)
  - many comments of the sort: *we did this before* …
  - main points: make it work, show that it works!
- in retrospective
  - classification considerations might have been useful since we tried to use SAT for symbolic model checking without taking Savitch's theorem into account
  - but might have prevented us going along that route ...

JKU
JOHANNES KEPLER
UNIVERSITÄT LINZ

- P
  - problems with polynonmially **time**-bounded algorithms
  - bounds measured in terms of input (file) size
- NP
  - same as P but with non-determinininistic choice
  - needs a SAT solver
- PSPACE
  - as P but **space**-bounded
  - QBF and bit-level model checking fall in this class
- NEXPTIME
  - same as NP but with exponential time
- P $\subseteq$ NP $\subseteq$ PSPACE $\subseteq$ NEXPTIME
  - usually it is assumed: P $\neq$ NP
  - it is further known: NP $\neq$ NEXPTIME



- NP problems
  - anything which can be (polynomially) encoded into SAT
  - combinational equivalence checking, bounded model checking
- PSPACE problems
  - anything which can be encoded (polynomially) into QBF
  - or into (bit-level) symbolic model checking
  - sequential equivalence checking, combinational synthesis or bounded games
- NEXPTIME problems
  - anything which can be encoded **exponentially** into SAT
  - first-order logic Bernays-Schönfinkel class (EPR): no functions, $\exists^*\forall^*$ prefix
  - QBF with explicit dependencies (Henkin Quantifiers): DQBF
  - partial observation games, black-box bounded model checking
  - bit-vector logics: QF_BV

- QF_BV contained in NEXPTIME
  - bit-blast (single exponentially)
  - give resulting formula to SAT solver
- we showed QF_BV is NEXPTIME hard by reducing DQBF to QF_BV

$$\forall x_0, x_1, x_2, x_3, x_4 \, \exists e_0(x_0, x_1, x_2, x_3), e_1(x_1, x_2, x_3, x_4) \; \varphi$$

  - polynomially encodes dependencies (for Henkin quantiers)
  - my student has now an (yet unpublished) direct proof
- why are bit-vectors NEXPTIME complete?

```
(set-logic QF_BV)
(declare-fun x () (_ BitVec 1000000))
(declare-fun y () (_ BitVec 1000000))
(declare-fun z () (_ BitVec 1000000))
(assert (= z (bvadd x y)))
(assert (= z (bvshl x (_ bv1 1000000))))
(assert (distinct x y))
```

$x, y : \text{bool}[1000000]$

$$y \neq x \;\wedge\; x + y = x \ll 1$$

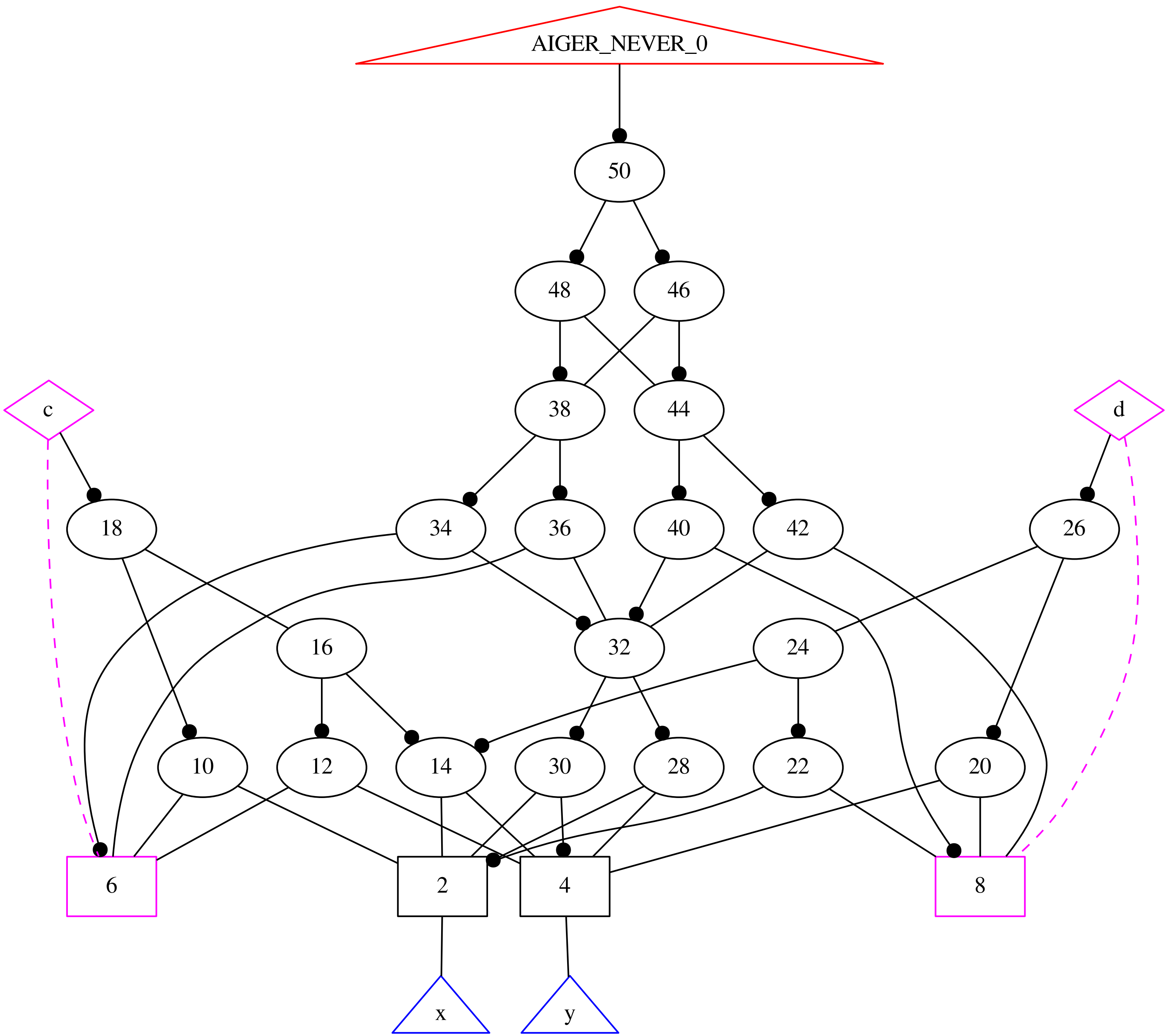
Bit-Wise Operators and Shifting Neighbouring Bits Only

- NP complete: QF_BV_{bw}
  - **relate <u>same</u> bits**: equality and all bit-wise operators
  - similar to well-known Ackermann reduction
- PSPACE complete: $\text{QF\_BV}_{bw,\ll 1}$
  - only allow operators which **relate <u>neighbouring</u> bits**:
    - base operators: equality, inequality/comparison, bit-wise ops, shift-by-one
    - extended operators: addition, multiplication by constants, single-bit-slices etc.
  - encode in symbolic model checking logarithmically in bit-width
- see our CSR'12, SMT'13 papers and our 2015 journal article in TOCS.
- came accross otherwise unsolvable benchmarks from industry!

Commutativity of Bit-Vector Addition in SMV


```
MODULE main
VAR
  c : boolean;     -- carry 'bvadd x y'
  d : boolean;     -- carry 'bvadd y x'
  x : boolean;     -- x0, x1, ...
  y : boolean;     -- y0, y1, ...
ASSIGN
  init (c) := FALSE;
  init (d) := FALSE;
ASSIGN
  next (c) := c & x | c & y | x & y;
  next (d) := d & y | d & x | y & x;
DEFINE
  o := c != (x != y);
  p := d != (y != x);
SPEC
  AG (o = p)
```

Commutativity of Bit-Vector Addition in AIGER



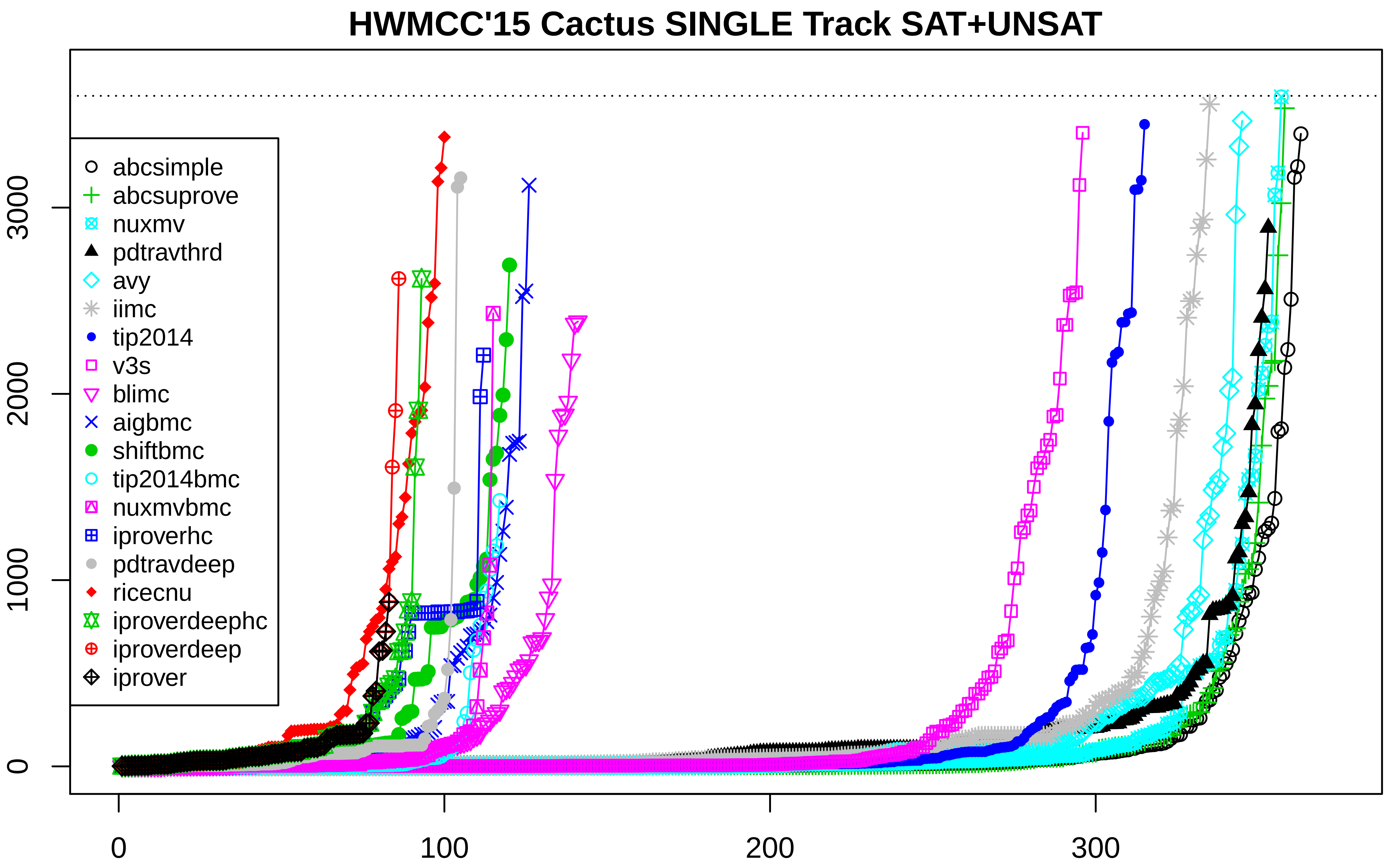
Hardware Model Checking Competition (HWMCC)

2006	2007	2008	2010	2011	2012	2013	2014	**2015**
AIGER format AVM'06 Ascona	1st HWMCC CAV'07 Berlin	2nd HWMCC CAV'08 Princeton	3rd HWMCC CAV'10 FLOC'10 Edinburgh	4th HWMCC FMCAD'11 Austin	5th HWMCC FMCAD'12 Cambridge UK	6th HWMCC FMCAD'13 Portland USA	7th HWMCC CAV'14 FLOC'14 Vienna Austria	**8th HWMCC FMCAD'15 Austin USA**
Founding Lunch CAV'06 FLOC'06 Seattle		HWMCC Lunch FMCAD'08 Portland						

- founding lunch at CAV'06, first competition at CAV'07
- HWMCC lunch at FMCAD'08 $\Rightarrow$ need multiple properties !!!
- affilliated with either CAV (7,8,10,14) or FMCAD (11,12,13,15)
- HWMCC'11: old SINGLE, new LIVEness and new MULTI property track
- HWMCC'12 as HWMCC'11, new DEEP bounds track sponsored by Oski
- in essence no change in HWMCC'12 - HWMCC'15
- HWMCC'15: DEEP, SINGLE, and LIVE, ~~MULTI~~, **1h time limit**, before 15min

20 Model Checkers in HWMCC'15

- **abcsimple**, **abcsimplive**, **abcsuprove** from Berkeley — Brayton,Sterin,Mishchenko,...
- **aigbmc**, **blimc** from JKU Linz — Biere
- **avy** from Technion+SEI+Princeton — Vizel,Gurfinkel,Malik
- **iimc** from Boulder — Somenzi,Bradley,Hassan
- **iprover(hc)**,**iproverdeep(hc)** from Manchester — Tsarkov,Korovin
- **nuxmv**,**nuxmvbmc** from Trento — Griggio,Roveri,...
- **pdtravdeep**, **pdtravthrd** from Torino — Cabodi,Quer,...
- **ricecnu** from Rice — Li,Vardi
- **shiftbmc** from Dresden — Manthey
- **tip2014**, **tip2014bmc** from Chalmers — Sörensson,Claessen
- **v3s** from Taipei — Yang,Wu,Huang


HWMCC'15 Cactus SINGLE Track SAT+UNSAT
0
1000
2000
3000
0
100
200
300
abcsimple
abcsuprove
nuxmv
pdtravthrd
avy
iimc
tip2014
v3s
blimc
aigbmc
shiftbmc
tip2014bmc
nuxmvbmc
iproverhc
pdtravdeep
ricecnu
iproverdeephc
iproverdeep
iprover

HWMCC'15 Table SINGLE SAT+UNSAT

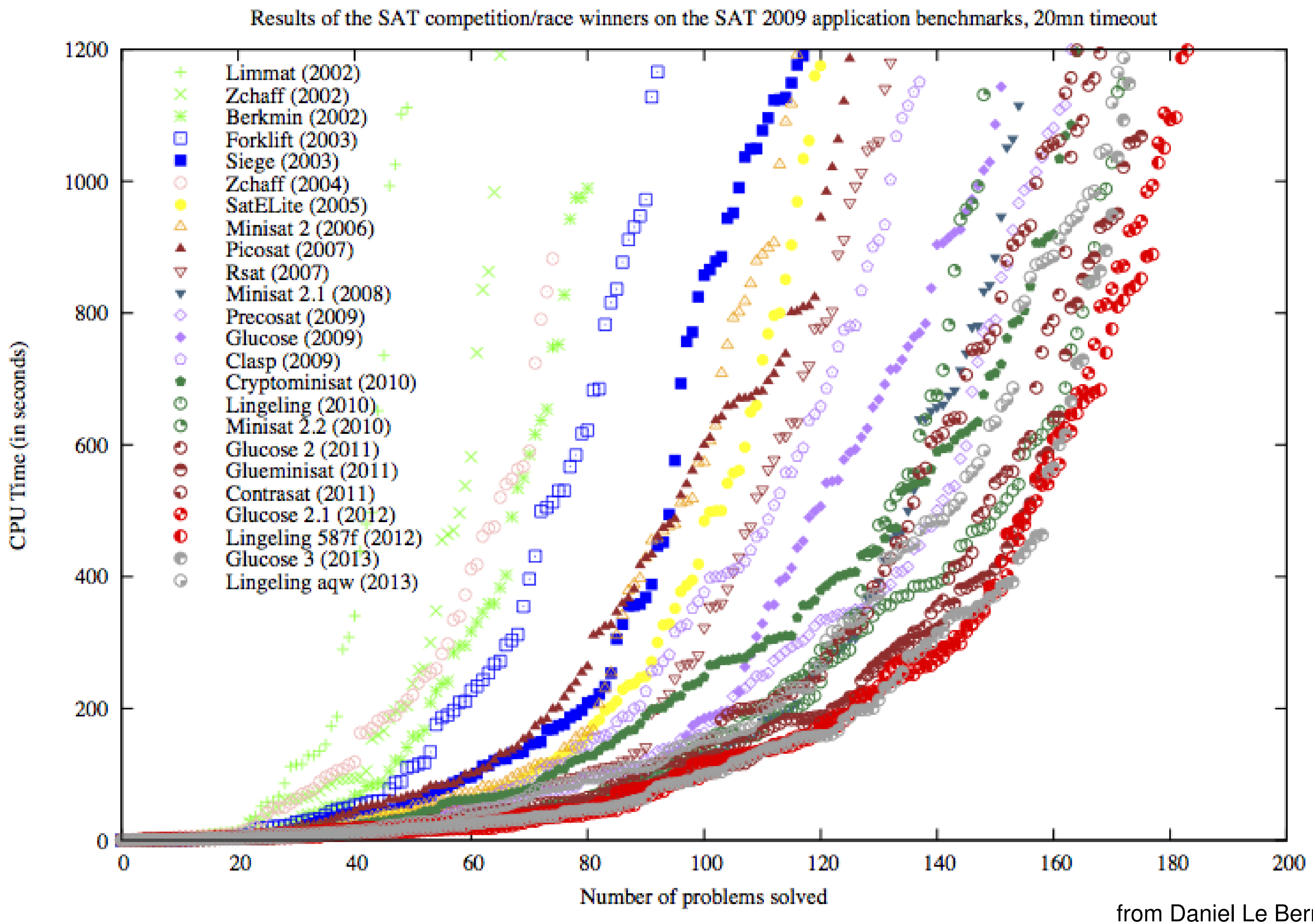
rank		cnt	ok	sat	uns	fld	to	mo	s11	s6	unk	real	time	max	best	uniq
1	abcsimple	548	363	122	241	185	154	31	0	0	0	45438	161407	6785	28	1
	abcsuprove	548	358	120	238	190	156	34	0	0	0	41596	115597	6905	23	1
2	nuxmv	548	357	127	230	191	165	26	0	0	0	47600	186902	6900	30	0
	pdtravthrd	548	353	115	238	195	62	124	3	0	6	52819	158951	6824	18	3
3	avy	548	345	115	230	203	171	30	0	0	2	49724	166878	5924	19	1
	iimc	548	335	105	230	213	163	50	0	0	0	57840	202386	6969	76	7
	tip2014	548	315	98	217	233	233	0	0	0	0	46817	46596	1062	144	0
	v3s	548	296	62	234	252	141	7	102	2	0	54273	152125	6037	28	4
	blimc	548	141	128	13	407	287	20	0	0	100	31687	31584	2241	61	1
	aigbmc	548	126	126	0	422	310	43	0	0	69	34210	34104	3644	47	1
	shiftbmc	548	120	120	0	428	247	12	0	0	169	37819	37697	1810	28	0
	tip2014bmc	548	117	117	0	431	225	50	0	0	156	11054	10911	5496	64	1
	nuxmvbmc	548	115	115	0	433	256	28	0	0	149	8885	8796	2504	45	0
	iproverhc	548	112	64	48	436	156	0	0	0	280	32269	17292	6730	3	0
	pdtravdeep	548	105	46	59	443	345	86	4	0	8	14725	14568	4202	11	0
	ricecnu	548	100	30	70	440	342	88	0	10	0	49153	49054	2204	0	0
	iproverdeephc	548	93	46	47	455	177	0	0	0	278	18116	17482	6728	2	0
	iproverdeep	548	86	46	40	462	149	0	0	0	313	14899	14818	6322	1	0
	iprover	548	83	43	40	465	59	0	0	0	406	8768	8698	6176	3	0

hors concours (not ranked):

aigbmc blimc: organizer model checkers
pdtravthrd: issue catching 'FATAL' for 'intel045' (not counted)
ricecnu: reports 8 instances SAT which are UNSAT (not counted)
iprover*hc: last minute (script) fixes after deadline

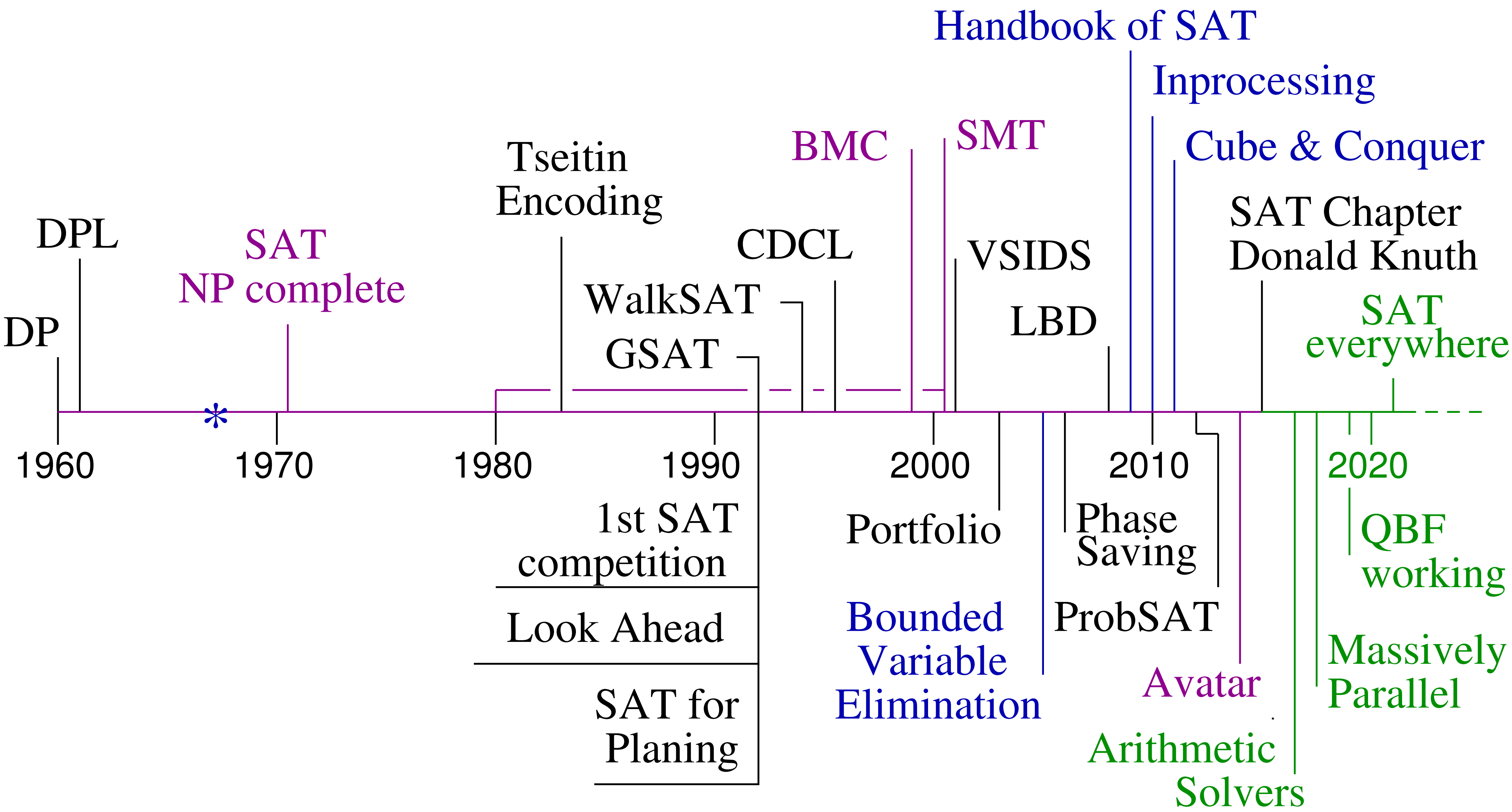
each team / submitter only ranked once (one medal maximum)


SAT-Race 2010 Award
"Plingeling"
by Armin Biere
is awarded the title of
Best Parallel Solver
Chair of SAT-Race Organizing Committee
SAT-Race 2010 Award
"Lingeling"
by Armin Biere
is awarded the title of
Edinburgh, Scotland, July 14, 2010
Tenth International Conference
On Theory and Applications of Satisfiability Testing
Competition 2007
Gold medal
The SAT2007 competition judges and organizers
Eighteenth International Conference on Theory and Applications of Satisfiability Testing
SAT-Race 2015
Main Track
Special Prize:
Most Innovative Solver
Awarded to:
Lingeling sr15bal
Written by:
Armin Biere
Parallel Track
3rd Prize
Awarded to:
Written by:
Armin Biere
2nd Prize
Twelfth International Conference
On Theory and Applications of Satisfiability Testing
SAT-Race 2010
Gold medal
Best Parallel Solver
2010
SAT COMPETITION 2013
Competition 2009
Silver medal
Winner
SAT Compe
Delft
University of Cincinnati



from Daniel Le Berre

Personal SAT Solver History



Satisfiability (SAT) related topics have attracted researchers from various disciplines. Logic, applied areas such as planning, scheduling, operations research and combinatorial optimization, but also theoretical issues on the theme of complexity, and much more, they all are connected through SAT.

My personal interest in SAT stems from actual solving: The increase in power of modern SAT solvers over the past 15 years has been phenomenal. It has become the key enabling technology in automated verification of both computer hardware and software. Bounded Model Checking (BMC) of computer hardware is now probably the most widely used model checking technique. The counterexamples that it finds are just satisfying instances of a Boolean formula obtained by unwinding to some fixed depth a sequential circuit and its specification in linear temporal logic. Extending model checking to software verification is a much more difficult problem on the frontier of current research. One promising approach for languages like C with finite word-length integers is to use the same idea as in BMC but with a decision procedure for the theory of bit-vectors instead of SAT. All decision procedures for bit-vectors that I am familiar with ultimately make use of a fast SAT solver to handle complex formulas.

Decision procedures for more complicated theories, like linear real and integer arithmetic, are also used in program verification. Most of them use powerful SAT solvers in an essential way.

Clearly, efficient SAT solving is a key technology for 21st century computer science. I expect this collection of papers on all theoretical and practical aspects of SAT solving will be extremely useful to both students and researchers and will lead to many further advances in the field.

Edmund Clarke

Edmund M. Clarke, FORE Systems University Professor of Computer Science and Professor of Electrical and Computer Engineering at Carnegie Mellon University, is one of the initiators and main contributors to the field of Model Checking, for which he also received the 2007 ACM Turing Award.

In the late 90s Professor Clarke was one of the first researchers to realize that SAT solving has the potential to become one of the most important technologies in model checking.

ISBN 978-1-58603-929-5
ISSN 0922-6389

185

HANDBOOK

of satisfiability

Editors:
Armin Biere
Marijn Heule
Hans van Maaren
Toby Walsh

IOS Press

Frontiers in Artificial Intelligence and Applications

HANDBOOK of satisfiability

Editors:
Armin Biere
Marijn Heule
Hans van Maaren
Toby Walsh

IOS Press

Part I. Theory and Algorithms

Part II. Applications and Extensions


P
9-8-06

File Edit View Document Tools Window Help X
5 / 318 150% Find

Special thanks are due to Armin Biere, Randy Bryant, Sam Buss, Niklas Eén, Ian Gent, Marijn Heule, Holger Hoos, Svante Janson, Peter Jeavons, Daniel Kroening, Oliver Kullmann, Massimo Lauria, Wes Pegden, Will Shortz, Carsten Sinz, Niklas Sörensson, Udo Wermuth, Ryan Williams, and . . . for their detailed comments on my early attempts at exposition, as well as to numerous other correspondents who have contributed crucial corrections. Thanks also to Stanford's Information Systems Laboratory for providing extra computer power when my laptop machine was inadequate.

* * *

Wow — Section 7.2.2.2 has turned out to be the longest section, by far, in *The Art of Computer Programming*. The SAT problem is evidently a "killer app," because it is key to the solution of so many other problems. Consequently I can only hope that my lengthy treatment does not also kill off my faithful readers! As I wrote this material, one topic always seemed to flow naturally into another, so there was no neat way to break this section up into separate subsections. (And anyway the format of *TAOCP* doesn't allow for a Section 7.2.2.2.1.)

I've tried to ameliorate the reader's navigation problem by adding subheadings at the top of each right-hand page. Furthermore, as in other sections, the exercises appear in an order that roughly parallels the order in which corresponding topics are taken up in the text. Numerous cross-references are provided

Biere
Bryant
Buss
Eén
Gent
Heule
Hoos
Janson
Jeavons
Kroening
Kullmann
Lauria
Pegden
Shortz
Sinz
Sörensson
Wermuth
Williams
Internet
MPR
Internet

8.26 x 11.69 in

- competitions are used to
  - compare and evaluate implementations and algorithms
  - generate benchmarks used in papers
- SAT competition is one of the largest competitions
  - many solvers, highly competitive
  - portfolio solving, over-tuning issues
  - benchmark selection scheme broken due to competing goals:
    - assess the state-of-the-art
    - high-light new ideas
    - give a fair chance to everybody
- research in SAT solving, verification, etc. in essence **empirical science**
  - benchmark selection critical
  - how to select benchmarks?
    - for the competition?
    - in your papers?

Conclusion

- what I did not talk about ... (yet)
  - parallel SAT
  - QBF / quantifiers in general
  - huge improvements in local research in recent years
  - how to apply local search to bit-vectors and SMT
  - testing / debugging
  - assertion synthesis
- acknowledgements:

  Ed Clarke, all co-authors, collaborators, students and Post-Docs

  and if would list more names I would struggle with order and probably forget somebody
- if you have model checking, SMT, or SAT problems you want share let me know . . .

looking for Post-Doc's and PhD students too