Fueling the SAT Revolution in Automated Reasoning

Armin Biere

50 Jahre KIT-Fakultät für Informatik

Karlsruhe Institut für Technologie

October 20, 2022

Biggest thing in Computer Science in the 21st century?

A _____ _____

Biggest thing in Computer Science in the 21st century!

Automated Reasoning

stole this joke from an invited talk of Clark Barrett

Seriously . . . `https://www.amazon.science/research-areas/automated-reasoning`

RESEARCH AREA

Automated reasoning

Focusing on the automation of formal logical reasoning to raise the bar on the security, durability, availability, and quality of Amazon's products and services.

Explore more

AWS IAM Access Analyzer | Formal methods | Formal verification | + 2 more

AUTOMATED REASONING

A billion SMT queries a day

CAV keynote lecture by the director of applied science for AWS Identity explains how AWS is making the power of automated reasoning available to all customers.

By Neha Rungta

August 18, 2022

Share

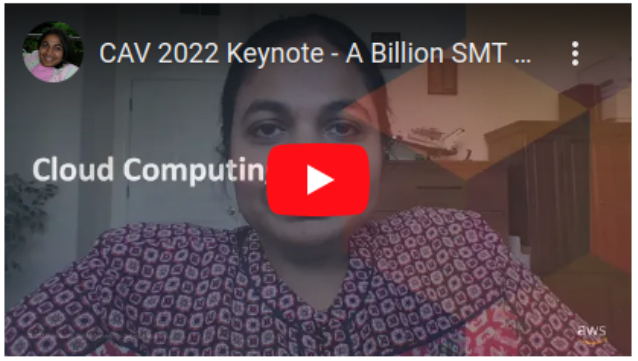
Conference

FLoC 2022

At this year's Computer-Aided Verification (CAV) conference — a leading automated-reasoning conference collocated with the Federated Logic Conferences (FLoC) — Amazon's Neha Rungta delivered a keynote talk in which she suggested that innovations at Amazon have "ushered in the golden age of automated reasoning".

Amazon scientists and engineers are using automated reasoning to prove the correctness of critical internal systems and to help customers prove the security of their cloud infrastructures. Many of these innovations are being driven by powerful reasoning engines called SMT solvers.

Satisfiability problems, or SAT, ask whether it's possible to assign variables true/false values that satisfy a set of

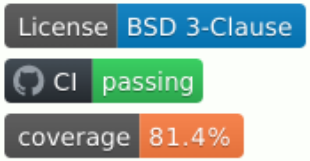


A billion SMT queries a day

Neha Rungta's 2022 CAV keynote

An efficient open-source automatic theorem prover for satisfiability modulo theories (SMT) problems.

Try cvc5 online!



People

Project Leaders

Clark Barrett (Stanford University)
Cesare Tinelli (University of Iowa)

Senior Technical Leads

Haniel Barbosa (Universidade Federal de Minas Gerais)
Aina Niemetz (Stanford University)
Mathias Preiner (Stanford University)
Andrew Reynolds (University of Iowa)

Current Developers

Martin Brain (University of Oxford and City University of London)
Hanna Lachnitt (Stanford University)
Abdalrhman Mohamed (University of Iowa)
Mudathir Mohamed (University of Iowa)
Alex Ozdemir (Stanford University)
Ying Sheng (Stanford University)
Yoni Zohar (Stanford University, Bar Ilan University)

Z3

An efficient SMT solver

Overview People Publications Downloads Groups Events News & features

Z3 is an efficient Satisfiability Modulo Theories (SMT) solver from Microsoft Research. Z3 is a solver for symbolic logic, a foundation for many software engineering tools. SMT solvers rely on a tight integration of specialized engines of proof. Each engine owns a piece of the global puzzle and implements specialized algorithms. For example, Z3's engine for arithmetic integrates Simplex, cuts and polynomial reasoning, while an engine for strings are regular expressions integrate methods for symbolic derivatives of regular languages. A theme shared among many of the algorithms is how they exploit a duality between finding satisfying solutions and finding refutation proofs. The solver also integrates engines for global and local inferences and global propagation. Z3 is used in a wide range of software engineering applications, ranging from program verification, compiler validation, testing, fuzzing using dynamic symbolic execution, model-based software development, network verification, and optimization.

Download from GitHub

Armin Biere
@ArminBiere

Hiking in Dagstuhl with @BjornerNikolaj

2:08 PM · Oct 12, 2022 · Twitter for Android

View Tweet analytics

30 Likes

Tweet your reply

Reply

Followed by some Tweeters you follow

Bach Le @BachLe13 · Oct 13

Replying to @ArminBiere and @BjornerNikolaj

Now SMT becomes the "backend" of SAT in this picture ...

1

09. – 14. Oktober 2022, Dagstuhl-Seminar 22411

Theory and Practice of SAT and Combinatorial Solving

Organisatoren

Olaf Beyersdorff (Friedrich-Schiller-Universität Jena, DE)
Armin Biere (Universität Freiburg, DE)
Vijay Ganesh (University of Waterloo, CA)
Jakob Nordström (University of Copenhagen, DK & Lund University, SE)

Workshops | Spring 2021

50 Years of Satisfiability: The Centrality of SAT in the Theory of Computing

(Weekly event) Feb 11, 2021 to May 13, 2021 (ended) *View all dates*

Program: Satisfiability: Theory, Practice, and Beyond

Add to Calendar

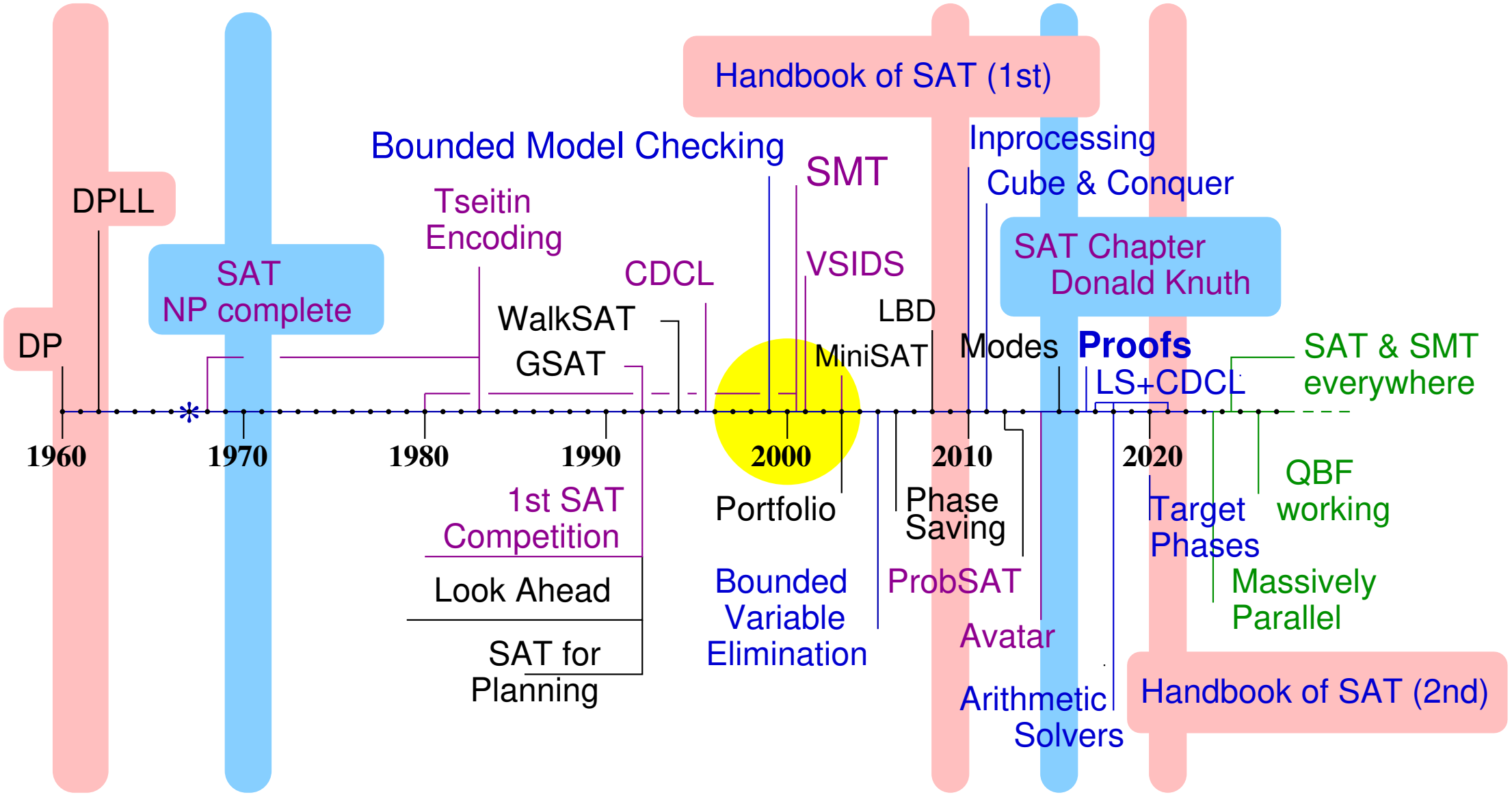
View schedule & video »

Organizers:

Albert Atserias (Universitat Politècnica de Catalunya; co-chair), Sam Buss (UC San Diego; co-chair), Paul Beame (University of Washington), Matti Järvisalo (University of Helsinki), Mohan Paturi (UC San Diego), Toni Pitassi (University of Toronto), Neil Thapen (Czech Academy of Sciences)

It has been nearly 50 years since the study of Satisfiability was initiated by the works of Cook, Karp and Levin. The ubiquity of NP-completeness caused the P versus NP question to become the defining central open question of Computational Complexity. This fundamental mathematical question remains open, but in the ensuing five decades, Satisfiability has become increasingly important in both theoretical and practical aspects of computer science. This workshop will explore and reflect on the role of Satisfiability in theoretical computer science: some talks will cover "big picture" topics, but most talks will emphasize current and recent research. Topics to be covered include: Proof

50 years of SAT NP-Complete



200 TB Biggest Math Proof Ever

HeuleKullmannMarek-SAT16 best paper

https://www.cs.utexas.edu/~marijn/ptn

Color the natural numbers $\mathbb{N}$ with two colors $\{\bullet, \bullet\}$, such that all pythagorean triples,

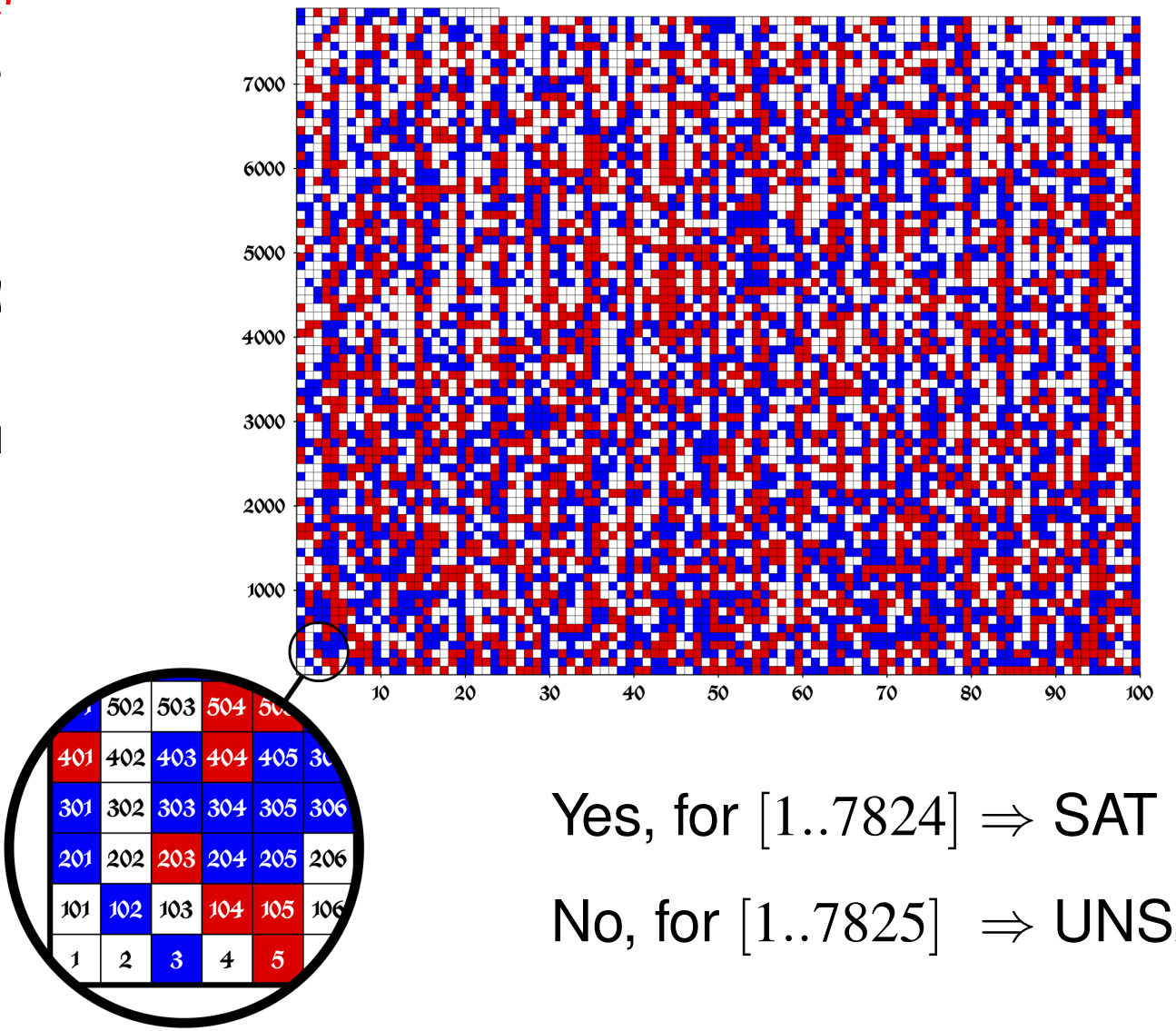
e.g., $3^2 + 4^2 = 5^2$

are NOT monochromatic?

have two colors

$(x_3 \vee x_4 \vee x_5) \wedge (\bar{x}_3 \vee \bar{x}_4 \vee \bar{x}_5)$
$(x_5 \vee x_{12} \vee x_{13}) \wedge (\bar{x}_5 \vee \bar{x}_{12} \vee$
$(x_6 \vee x_8 \vee x_{10}) \wedge (\bar{x}_6 \vee \bar{x}_8 \vee \bar{x}_1$
...

```
p cnf 7820 18930
3 4 5 0
-3 -4 -5 0
5 12 13 0
-5 -12 -13 0
...
5412 5635 7813 0
-5412 -5635 -7813 0
5474 5520 7774 0
-5474 -5520 -7774 0
```



Yes, for $[1..7824] \Rightarrow$ SAT

No, for $[1..7825] \Rightarrow$ UNSAT

```
void encode () {
  int n = 7825;
  for (int i = 1; i <= n; i++)
    for (int j = i; j <= n; j++) {
      int k = sqrt (i*i + j*j);
      if (k <= n && i*i + j*j == k*k)
        printf ("%d %d %d 0\n", i, j, k),
        printf ("%d %d %d 0\n", -i, -j, -k);
    }
}
```

Published: 26 May 2016

Two-hundred-terabyte maths proof is largest ever

Evelyn Lamb

Nature **534**, 17–18 (2016) | Cite this article

A computer cracks the Boolean Pythagorean triples problem – but is it really maths?

The University of Texas's Stampede supercomputer, on which the 200-terabyte maths proof was solved. Credit: University of Texas

Three computer scientists have announced the largest-ever mathematics proof: a file that comes in at a whopping 200 terabytes[1], roughly equivalent to all the digitized text held by the US Library of Congress. The researchers have created a 68-gigabyte compressed version of their solution – which would allow anyone with about 30,000 hours of spare processor time to download, reconstruct and verify it – but a human could never hope to read through it.

REVIEW ARTICLES

The Science of Brute Force

By Marijn J. H. Heule, Oliver Kullmann
Communications of the ACM, August 2017, Vol. 60 No. 8, Pages 70-79
10.1145/3107239
Comments (1)

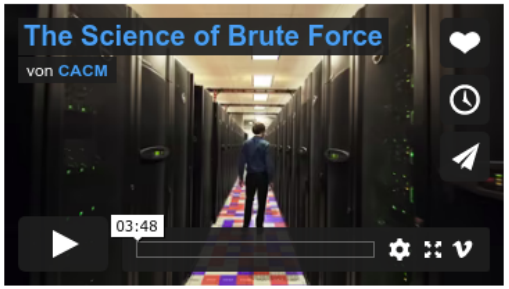
Credit: Peter Crowther Associates

Recent progress in automated reasoning and super-computing gives rise to a new era of brute force. The game changer is "SAT," a disruptive, brute-reasoning technology in industry and science. We illustrate its strength and potential via the proof of the Boolean Pythagorean Triples Problem, a long-standing open problem in Ramsey Theory. This 200TB proof has been constructed completely automaticallyparadoxically, in an ingenious way. We welcome these bold new proofs emerging on the horizon, beyond human understandingboth mathematics and industry need them.

Back to Top

Key Insights

- Long-standing open problems in mathematics can now be solved completely automatically resulting in clever though potentially gigantic proofs.
- Our time requires answers to hard questions regarding safety and security. In these cases knowledge is more important than understanding as long as we can trust the answers.
- Powerful SAT-solving heuristics facilitate linear speedups even when using thousands of cores. Combined with the ever-increasing capabilities of high-performance computing clusters they enable solving challenging problems.

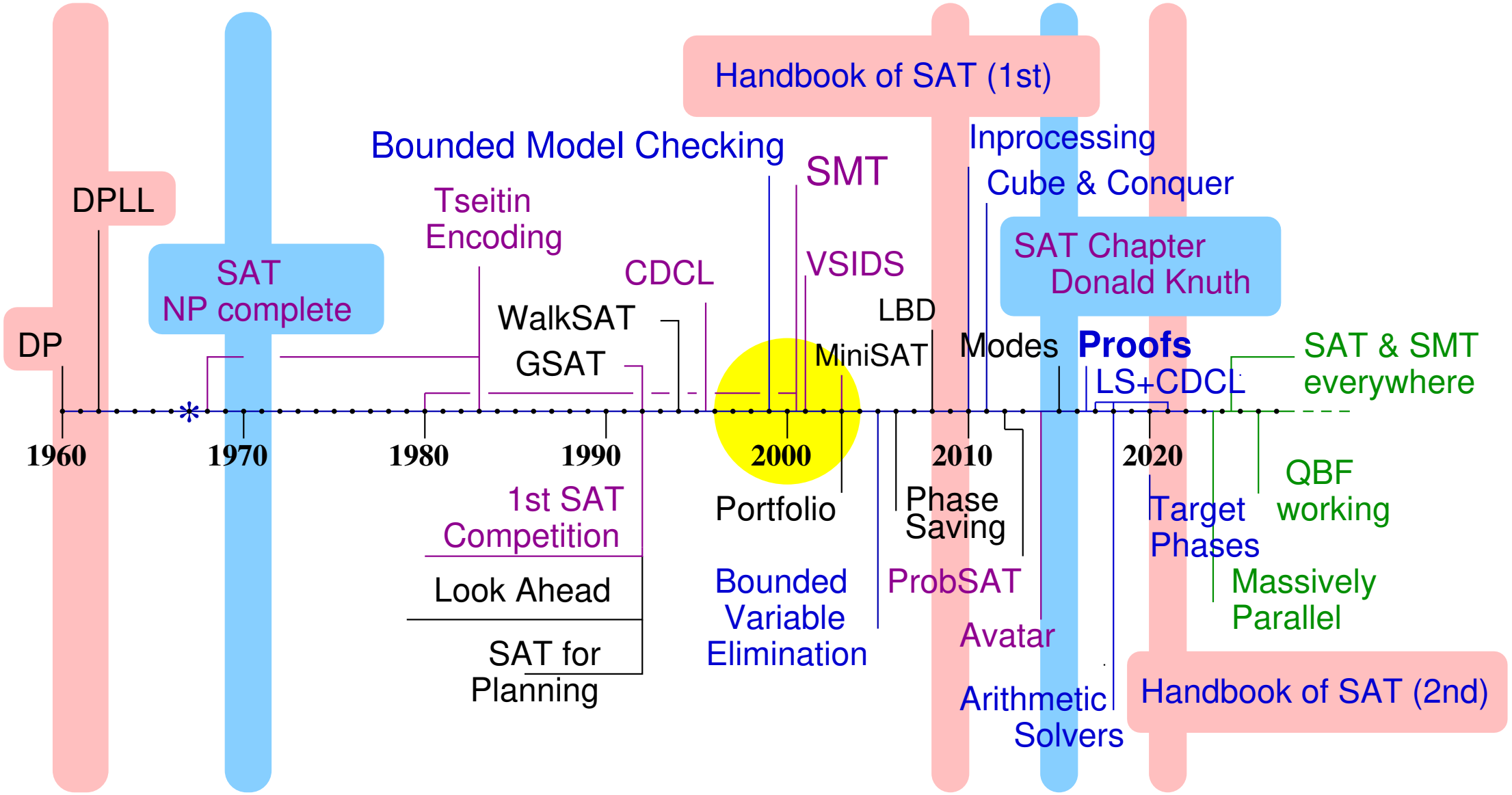


Many relevant search problems, from artificial intelligence to combinatorics, explore large search spaces to determine the presence or absence of a certain object. These problems are hard due to combinatorial explosion, and have traditionally been called infeasible. The brute-force method, which at least implicitly explores all possibilities, is a general approach to systematically search through such spaces.

Brute force has long been regarded as suitable only for simple problems. This has changed in the last two decades, due to the progress in Satisfiability (SAT) solving, which by adding brute reason renders brute force into a powerful approach to deal with many problems easily and automatically. Search spaces with far more possibilities than the number of particles in the universe may be completely explored.

MORE NEWS & OPINIONS
Congress Mandates Technologies to Stop Drunk Driving
Associated Press
How AI Is Reinventing What Computers Are
MIT Technology Review
What Should be Done About Facebook?
Jason Hong

Handbook'09


Handbook of SAT (1st)
Bounded Model Checking
SMT
Inprocessing
Cube & Conquer
DPLL
Tseitin
Encoding
SAT Chapter
Donald Knuth
SAT
NP complete
CDCL
VSIDS
DP
WalkSAT
LBD
GSAT
MiniSAT
Modes
Proofs
SAT & SMT
everywhere
LS+CDCL
1960
1970
1980
1990
2000
2010
2020
QBF
working
1st SAT
Competition
Portfolio
Phase
Saving
Target
Phases
Look Ahead
Bounded
Variable
Elimination
ProbSAT
Massively
Parallel
Avatar
SAT for
Planning
Arithmetic
Solvers
Handbook of SAT (2nd)

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

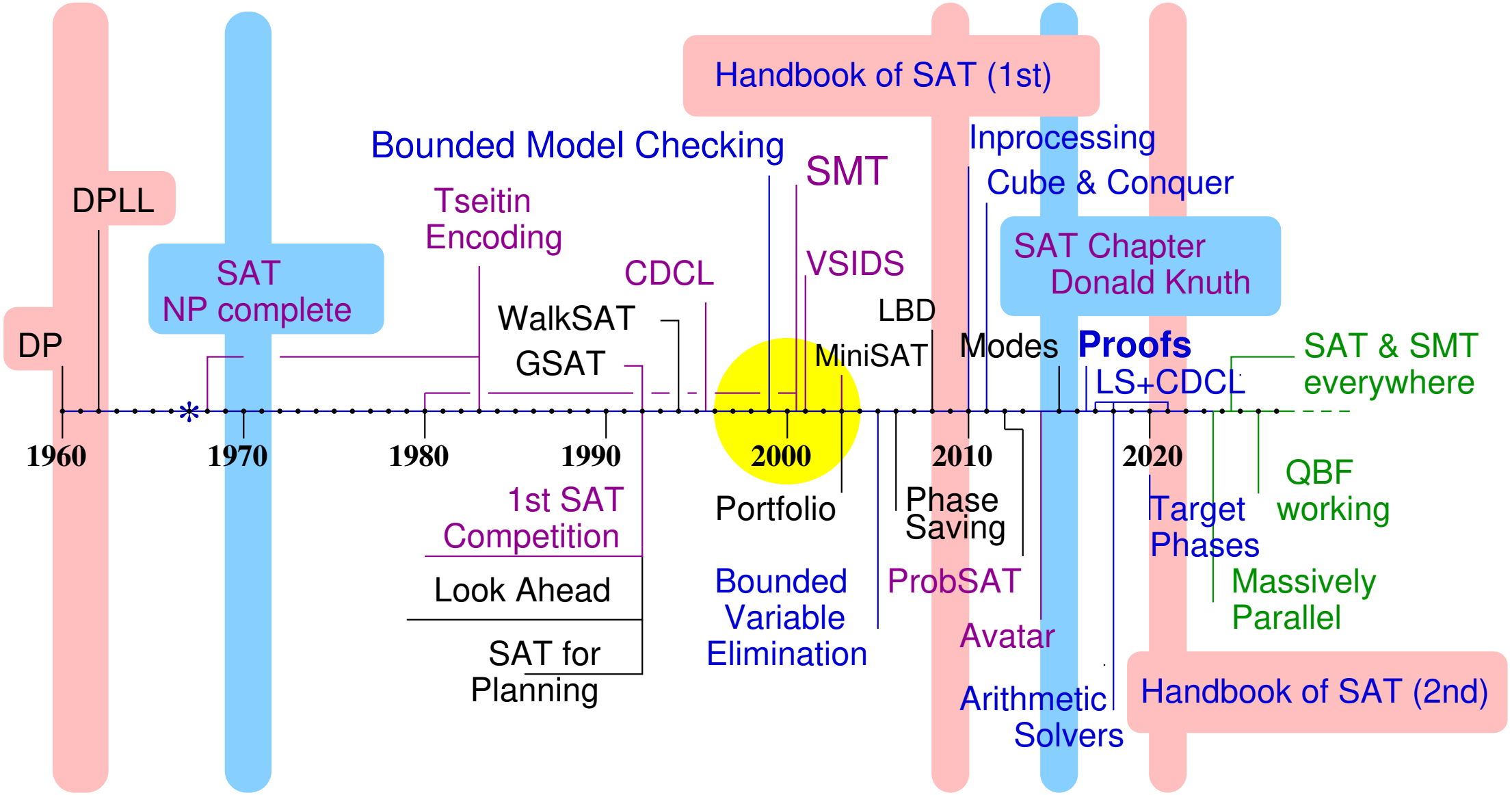
Editors:
Armin Biere
Marijn Heule
Hans van Maaren
Toby Walsh

IOS Press

Frontiers in Artificial Intelligence and Applications
HANDBOOK
of satisfiability
Editors:
Armin Biere
Marijn Heule
Hans van Maaren
Toby Walsh
IOS Press

SAT Handbook 1st Edition (2009)

Knuth TAOCP Volume 4b Section 7.2.2.2 (300 pages)



NEWLY AVAILABLE SECTION OF
THE CLASSIC WORK

The Art of Computer Programming

VOLUME 4

Satisfiability

FASCICLE 6

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.

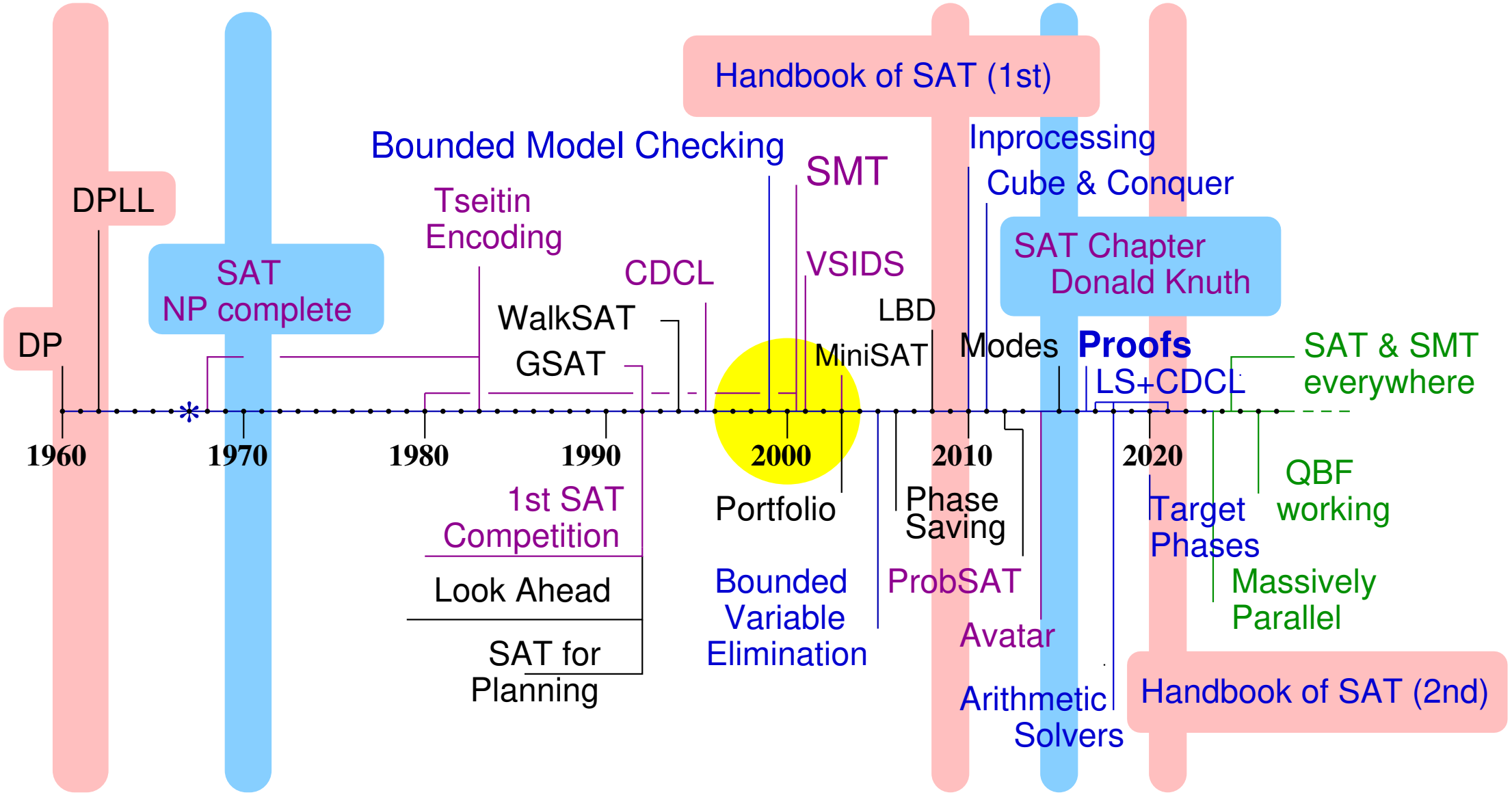
Biere
Bryant
Buss
Eén
Gent
Heule
Hoos
Janson
Jeavons
Kroening
Kullmann
Lauria
Pegden
Shortz
Sinz
Sörensson
Wermuth
Williams
Internet
MPR
Internet

* * *

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

Handbook'21


Handbook of SAT (1st)
Bounded Model Checking
SMT
Inprocessing
Cube & Conquer
DPLL
Tseitin
Encoding
SAT Chapter
Donald Knuth
SAT
NP complete
CDCL
VSIDS
DP
WalkSAT
LBD
GSAT
MiniSAT
Modes
Proofs
SAT & SMT
everywhere
LS+CDCL
1960
1970
1980
1990
2000
2010
2020
QBF
working
1st SAT
Competition
Portfolio
Phase
Saving
Target
Phases
Look Ahead
Bounded
Variable
Elimination
ProbSAT
Massively
Parallel
Avatar
SAT for
Planning
Arithmetic
Solvers
Handbook of SAT (2nd)

SAT Handbook 2nd Edition (2021)

editors Armin Biere, Marijn Heule, Hans van Maaren, Toby Walsh

SAT solving is a key technology for 21st entury computer science.

Edmund Clarke
2007 ACM Turing Award Recipient

*with many updated chapters and the **following 7 new chapters**:*

Proof Complexity Jakob Nordström and Sam Buss

Preprocessing Armin Biere, Matti Järvisalo and Benjamin Kiesl

The SAT problem is evidently a killer app, because it is key to the solution of so many other problems.

Donald Knuth
1974 ACM Turing Award Recipient

Tuning and Configuration
Holger Hoos, Frank Hutter and Kevin Leyton-Brown

Proofs of Unsatisfiability Marijn Heule

The SAT problem is at the core of arguably the most fundamental question in computer science: What makes a problem hard?

Stephen Cook
1982 ACM Turing Award Recipient

Core-Based MaxSAT
Fahiem Bacchus, Matti Järvisalo and Ruben Martins

Proof Systems for Quantified Boolean Formulas
Olaf Beyersdorff, Mikoláš Janota, Florian Lonsing and Martina Seidl

Approximate Model Counting Supratik Chakraborty, Kuldeep S. Meel, and Moshe Y. Vardi

Frontiers in Artificial Intelligence and Applications

HANDBOOK of satisfiability

IOS Press

IOS Press Book

www.iospress.com

February 2021
Approx. 1516 pp.

Print book
Hardcover, in 2 parts
ISBN: 978-1-64368-160-3 (print)
€200 / US$250 / £180
excl. VAT

Ebook
ISBN: 978-1-64368-161-0 (online)
€200 / US$250 / £180
excl. VAT

Discount Code
Order your print book before April 15, 2021 and get 35% off!
Code: **SAT2021**

–35%

For more information and ordering check
tiny.cc/SAT2021

Handbook of Satisfiability

Second Edition

Editors: A. Biere, M. Heule, H. van Maaren, T. Walsh
Volume 336 of Frontiers in Artificial Intelligence and Applications

Propositional logic has been recognized throughout the centuries as one of the cornerstones of reasoning in philosophy and mathematics. Over time, its formalization into Boolean algebra was accompanied by the recognition that a wide range of combinatorial problems can be expressed as propositional satisfiability (SAT) problems. Because of this dual role, SAT developed into a mature, multi-faceted scientific discipline, and from the earliest days of computing a search was underway to discover how to solve SAT problems in an automated fashion.

This book, the *Handbook of Satisfiability*, is the second, updated and revised edition of the book first published in 2009 under the same name. The handbook aims to capture the full breadth and depth of SAT and to bring together significant progress and advances in automated solving. Topics covered span practical and theoretical research on SAT and its applications and include search algorithms, heuristics, analysis of algorithms, hard instances, randomized formulae, problem encodings, industrial applications, solvers, simplifiers, tools, case studies and empirical results. SAT is interpreted in a broad sense, so as well as propositional satisfiability, there are chapters covering the domain of quantified Boolean formulae (QBF), constraints programming techniques (CSP) for word-level problems and their propositional encoding, and satisfiability modulo theories (SMT). An extensive bibliography completes each chapter.

This second edition of the handbook will be of interest to researchers, graduate students, final-year undergraduates, and practitioners using or contributing to SAT, and will provide both an inspiration and a rich resource for their work.

Edmund Clarke, 2007 ACM Turing Award Recipient: "SAT solving is a key technology for 21st century computer science."

Donald Knuth, 1974 ACM Turing Award Recipient: "SAT is evidently a killer app, because it is key to the solution of so many other problems."

Stephen Cook, 1982 ACM Turing Award Recipient: "The SAT problem is at the core of arguably the most fundamental question in computer science: What makes a problem hard?"

IOS Press
Nieuwe Hemweg 6B
1013 BG Amsterdam
The Netherlands
Tel.: +31 20 688 3355
Email: order@iospress.nl

Visit our website www.iospress.com for online ordering
For ebooks, go to www.ebooks.iospress.com
Follow us on Twitter: @IOSPress_STM
Follow us on Facebook: publisheriospress

IOS Press, Inc.
6751 Tepper Drive
Clifton, VA 20124
USA
Tel.: +1 703 830 6300
Email: sales@iospress.com

Contents

IOS Press
Nieuwe Hemweg 6B
1013 BG Amsterdam
The Netherlands
Tel.: +31 20 688 3355
Email: order@iospress.nl

Visit our website www.iospress.com for online ordering
For ebooks, go to www.ebooks.iospress.com
Follow us on Twitter: @IOSPress_STM
Follow us on Facebook: publisheriospress

IOS Press, Inc.
6751 Tepper Drive
Clifton, VA 20124
USA
Tel.: +1 703 830 6300
Email: sales@iospress.com

Knuth

SEMINUMERICAL ALGORITHMS

THE ART OF COMPUTER PROGRAMMING

VOL. 2
SECOND EDITION

ADDISON WESLEY

3822

Knuth

FUNDAMENTAL ALGORITHMS

THE ART OF COMPUTER PROGRAMMING

VOL. 1
SECOND EDITION

ADDISON WESLEY

3809

The TEXbook

KNUTH

ADDISON WESLEY

13448

Updated and Revised

KNUTH

The Art of Computer Programming

Fundamental Algorithms

VOLUME 1
Third Edition

Addison Wesley

Updated and Revised

KNUTH

The Art of Computer Programming

Seminumerical Algorithms

VOLUME 2
Third Edition

Addison Wesley

Updated and Revised

KNUTH

The Art of Computer Programming

Sorting and Searching

VOLUME 3
Second Edition

Addison Wesley

Extended and Refined

KNUTH

The Art of Computer Programming

Combinatorial Algorithms, Part 1

VOLUME 4A

Addison Wesley

New

KNUTH

The Art of Computer Programming Volume 4 Fascicle 5

V4 F5

AW

New

KNUTH

The Art of Computer Programming Volume 4 Fascicle 6

V4 F6

AW

KERNIGHAN · RITCHIE

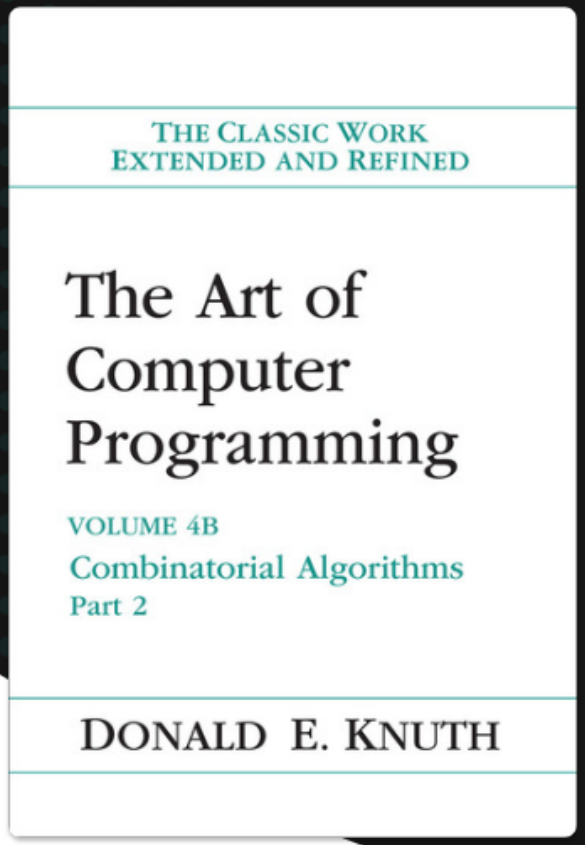
THE C PROGRAMMING LANGUAGE

SECOND EDITION

PTR PH

Kernighan/Ritchie · Programmieren in C

PC professionell



Art of Computer Programming, Volume 4B, The: Combinatorial Algorithms, 1st edition

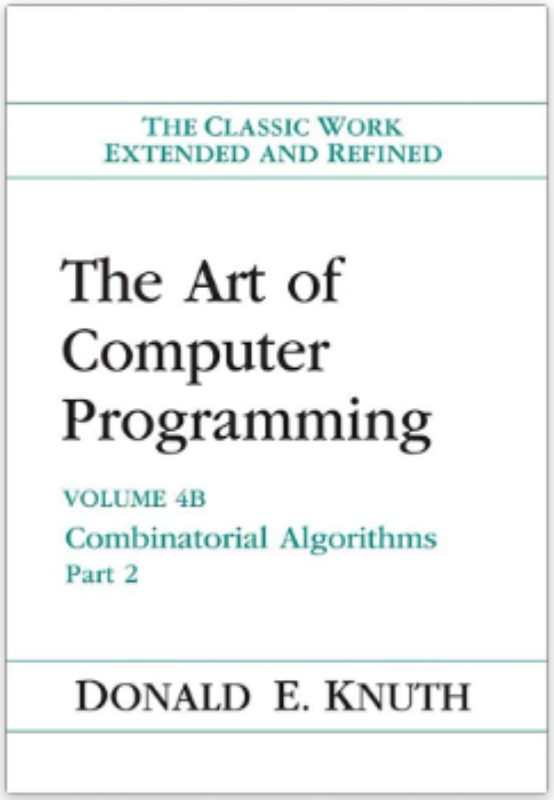
Published by Addison-Wesley Professional (September 28th 2022) - Copyright © 2023

Donald E. Knuth

ISBN-13: 9780201038064

Art of Computer Programming, The: Combinatorial Algorithms, Volume 4B

Blick ins Buch



The Art of Computer Programming, Volume 4B: Combinatorial Algorithms Gebundene Ausgabe – 21. Oktober 2022

Englisch Ausgabe | von Donald E. Knuth (Autor)

Buch 4 von 4: The Art of Computer Programming

Alle Formate und Editionen anzeigen

Kindle	Gebundenes Buch
10,29 €	**97,53 €**

Lesen Sie mit unserer **kostenfreien App**

2 Neu ab 82,15 €

The Art of Computer Programming is Knuth's multivolume analysis of algorithms. With the addition of this new volume, it continues to be the definitive description of classical computer science.

Volume 4B, the sequel to Volume 4A, extends Knuth's exploration of combinatorial algorithms. These algorithms are of keen interest to software designers because ". . . a single

Mehr lesen

Falsche Produktinformationen melden

Dieses Bild anzeigen

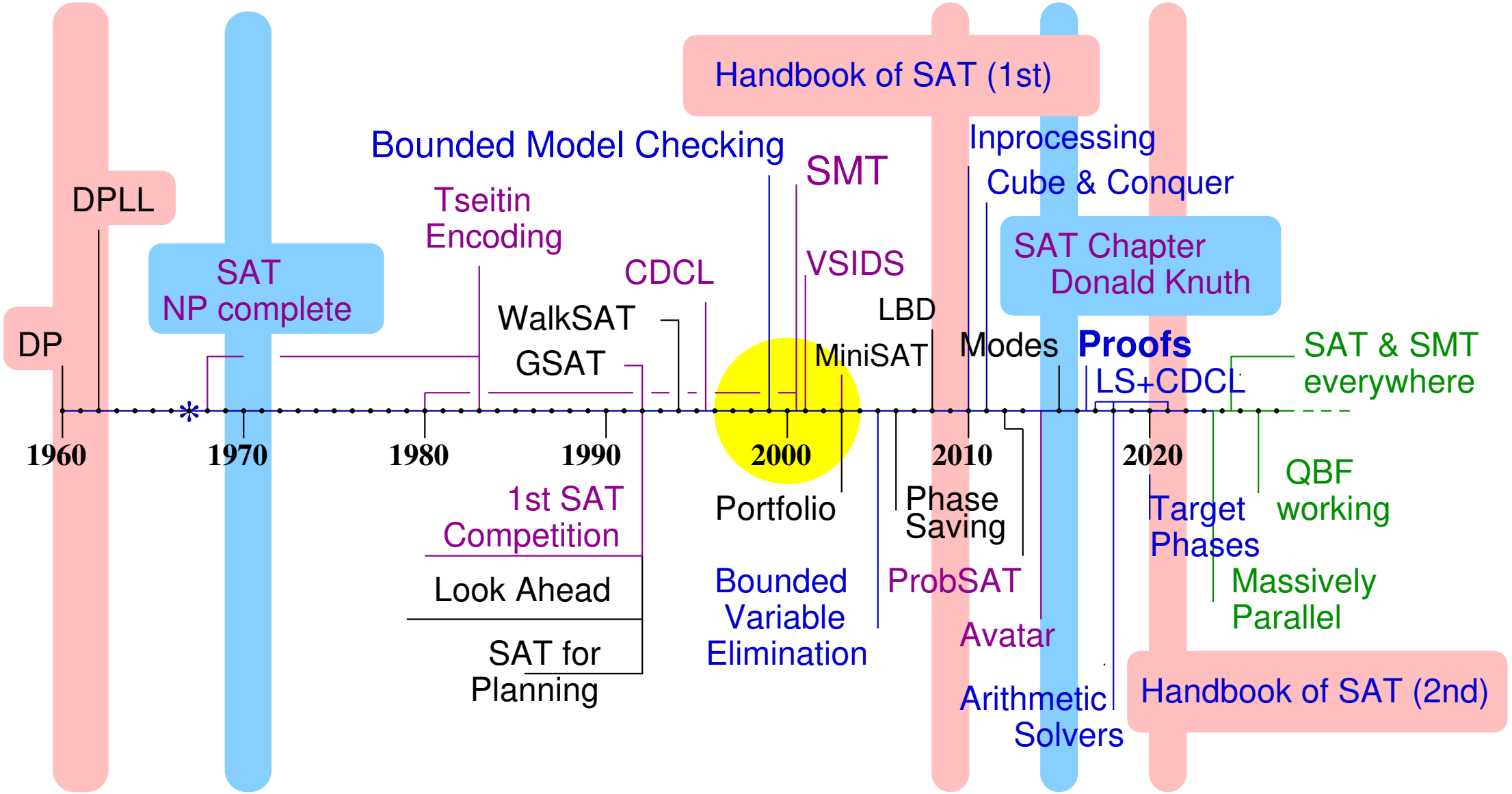
Dem Autor folgen

Donald E. Knuth

Folgen

Seitenzahl der Print-Ausgabe	Sprache	Herausgeber	Erscheinungstermin
640 Seiten	**Englisch**	**Addison Wesley**	**21. Oktober 2022**

60 Years of SAT Solving



DPLL from the 60ies

[DaviesPutnam'62][DavisLogemannLoveland'62]

$DPLL(F)$

$F := BCP(F)$ boolean constraint propagation

if $F = \top$ **return** satisfiable

if $\bot \in F$ **return** unsatisfiable

pick remaining variable x and literal $l \in \{x, \neg x\}$

if $DPLL(F \wedge \{l\})$ returns satisfiable **return** satisfiable

return $DPLL(F \wedge \{\neg l\})$

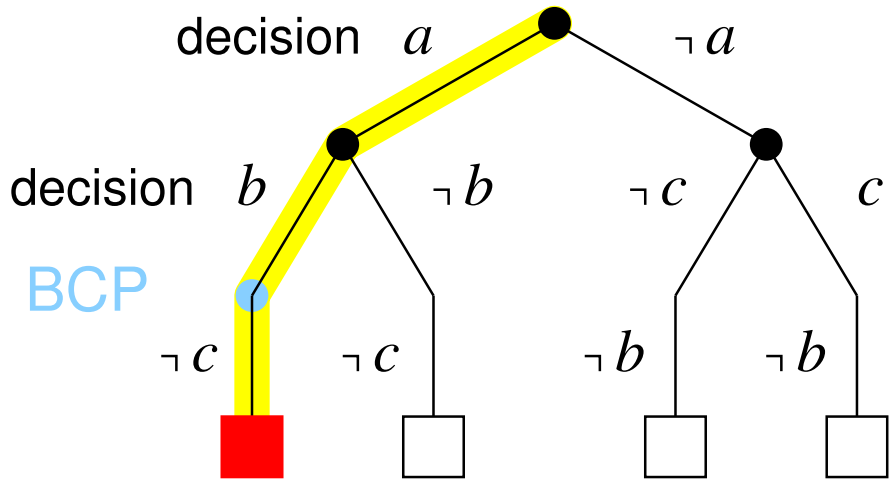
$\neq$ CDCL

DPLL Example

$a = 1$

$b = 1$

$c = 0$



clauses

$\neg a \vee \neg b \vee \neg c$

$\neg a \vee \neg b \vee c$

$\neg a \vee b \vee \neg c$

$\neg a \vee b \vee c$

$a \vee \neg b \vee \neg c$

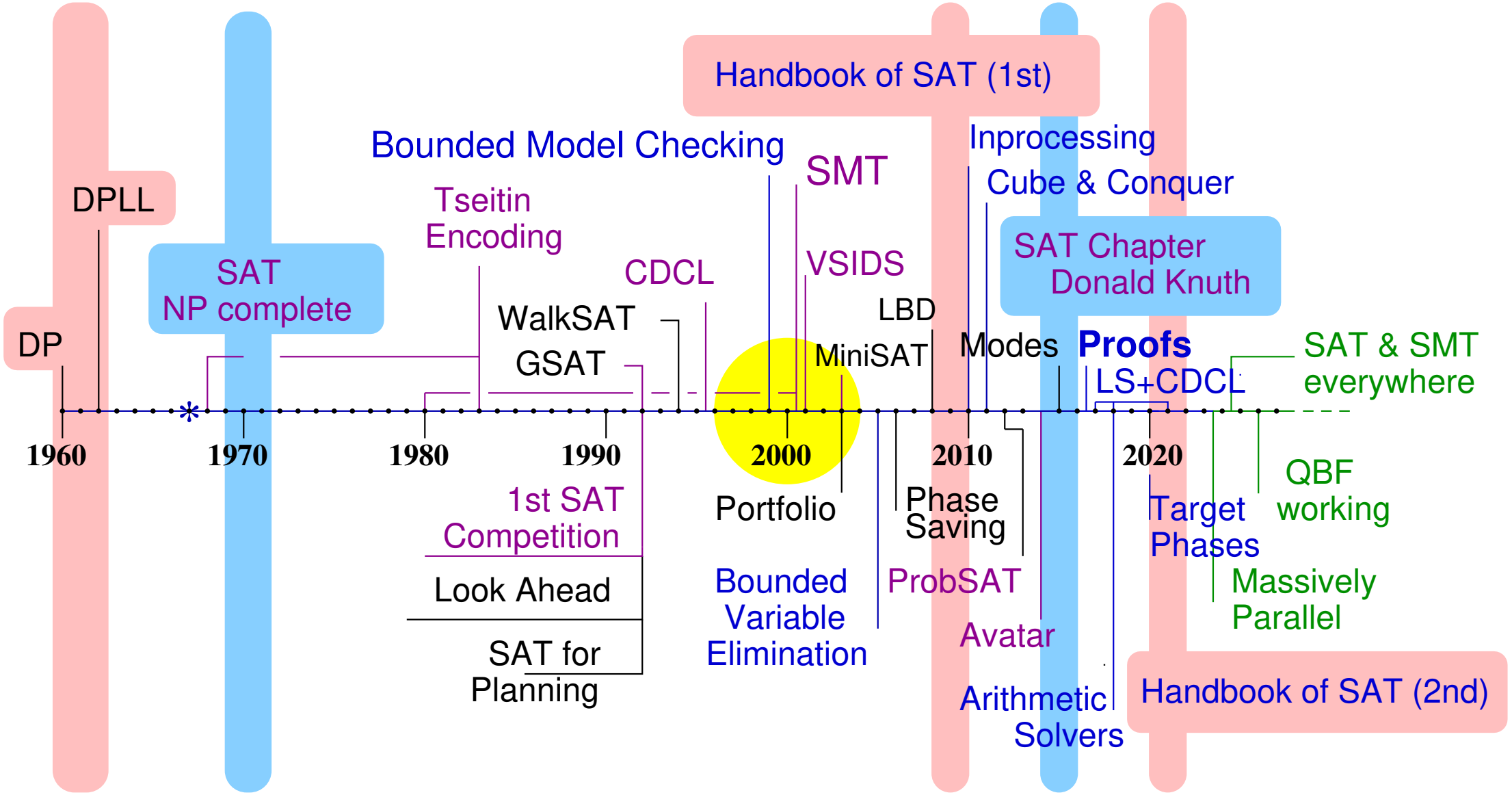
$a \vee \neg b \vee c$

$a \vee b \vee \neg c$

$a \vee b \vee c$

CDCL from the 90ies

[MarquesSilvaSakallah'96]

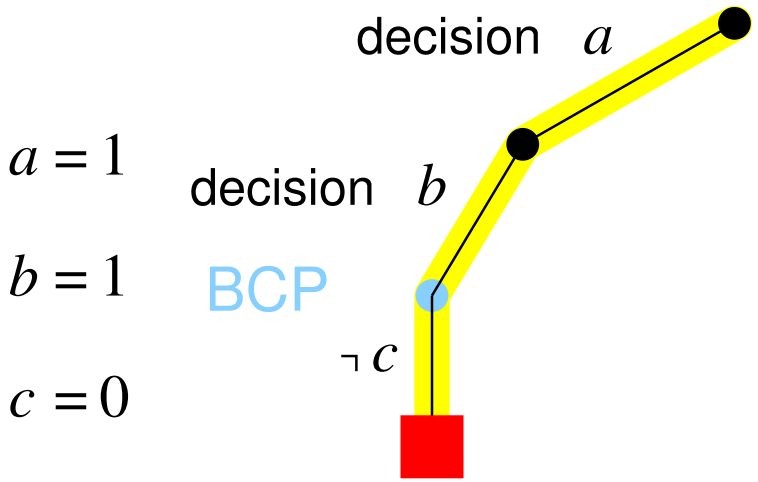


CDCL from the 90ies

[MarquesSilvaSakallah'96]

```
int basic_cdcl_loop () {
  int res = 0;

  while (!res)
        if (unsat) res = 20;
    else if (!propagate ()) analyze ();     // analyze propagated conflict
    else if (satisfied ()) res = 10;        // all variables satisfied
    else decide ();                         // otherwise pick next decision

  return res;
}
```

CDCL Example



clauses

$\neg a \vee \neg b \vee \neg c$

$\neg a \vee \neg b \vee c$

$\neg a \vee b \vee \neg c$

$\neg a \vee b \vee c$

$a \vee \neg b \vee \neg c$

$a \vee \neg b \vee c$

$a \vee b \vee \neg c$

$a \vee b \vee c$

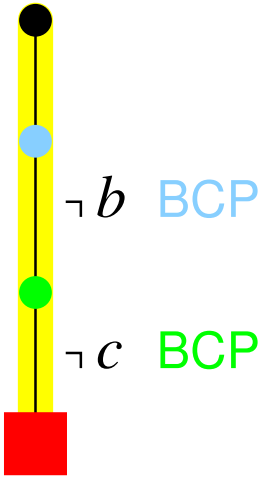
learn $\neg a \vee \neg b$

CDCL Example

$a = 1$

$b = 0$

$c = 0$



clauses

$\neg a \vee \neg b \vee \neg c$

$\neg a \vee \neg b \vee c$

$\neg a \vee b \vee \neg c$

$\neg a \vee b \vee c$

$a \vee \neg b \vee \neg c$

$a \vee \neg b \vee c$

$a \vee b \vee \neg c$

$a \vee b \vee c$

$\neg a \vee \neg b$

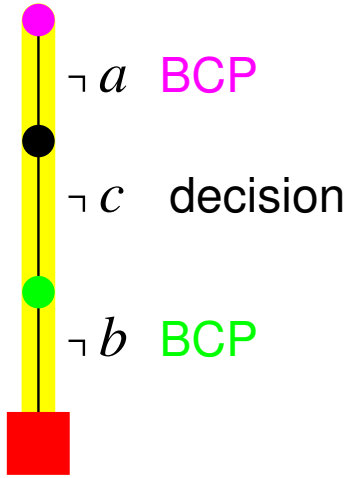
learn $\neg a$

CDCL Example

$a = 1$

$b = 0$

$c = 0$



clauses

$\neg a \vee \neg b \vee \neg c$

$\neg a \vee \neg b \vee c$

$\neg a \vee b \vee \neg c$

$\neg a \vee b \vee c$

$a \vee \neg b \vee \neg c$

$a \vee \neg b \vee c$

$a \vee b \vee \neg c$

$a \vee b \vee c$

$\neg a \vee \neg b$

$\neg a$

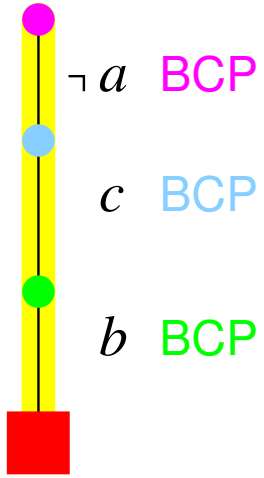
learn c

CDCL Example

$a = 1$

$b = 0$

$c = 0$



clauses

$\neg a \vee \neg b \vee \neg c$

$\neg a \vee \neg b \vee c$

$\neg a \vee b \vee \neg c$

$\neg a \vee b \vee c$

$a \vee \neg b \vee \neg c$

$a \vee \neg b \vee c$

$a \vee b \vee \neg c$

$a \vee b \vee c$

$\neg a \vee \neg b$

$\neg a$

c

learn $\bot$

empty clause

More Advanced CDCL Loop with Reduce and Restart

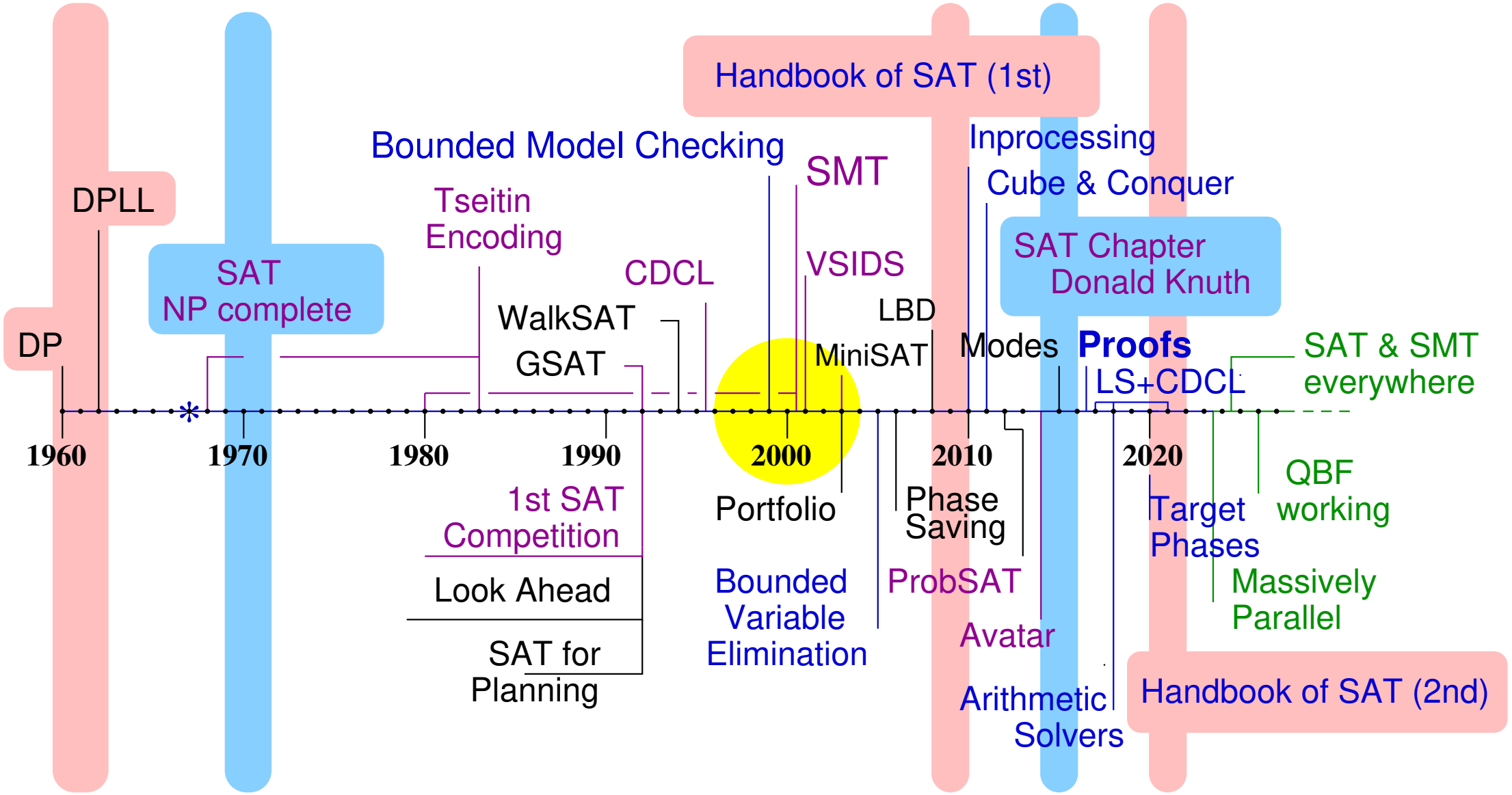
```
int basic_cdcl_loop_with_reduce_and_restart () {

  int res = 0;

  while (!res)
        if (unsat) res = 20;
    else if (!propagate ()) analyze ();    // analyze propagated conflict
    else if (satisfied ()) res = 10;       // all variables satisfied
    else if (restarting ()) restart ();    // restart by backtracking
    else if (reducing ()) reduce ();       // collect useless learned clauses
    else decide ();                        // otherwise pick next decision

  return res;
}
```

Inprocessing CaDiCaL CDCL Loop

https://github.com:arminbiere/cadical

```
while (!res) {
        if (unsat) res = 20;
  else if (unsat_constraint) res = 20;
  else if (!propagate ()) analyze ();      // propagate and analyze
  else if (iterating) iterate ();          // report learned unit
  else if (satisfied ()) res = 10;         // found model
  else if (search_limits_hit ()) break;    // decision or conflict limit
  else if (terminated_asynchronously ())    // externally terminated
    break;
  else if (restarting ()) restart ();      // restart by backtracking
  else if (rephasing ()) rephase ();       // reset variable phases
  else if (reducing ()) reduce ();         // collect useless clauses
  else if (probing ()) probe ();           // failed literal probing
  else if (subsuming ()) subsume ();       // subsumption algorithm
  else if (eliminating ()) elim ();        // variable elimination
  else if (compacting ()) compact ();      // collect variables
  else if (conditioning ()) condition ();  // globally blocked clauses
  else res = decide ();                    // next decision
}
```

Probabilistic local search SAT solvers WalkSAT'94 ProbSAT'12



WalkSAT'94 [SelmanKautzCohen'94] [McAllesterSelmanKautz'97]

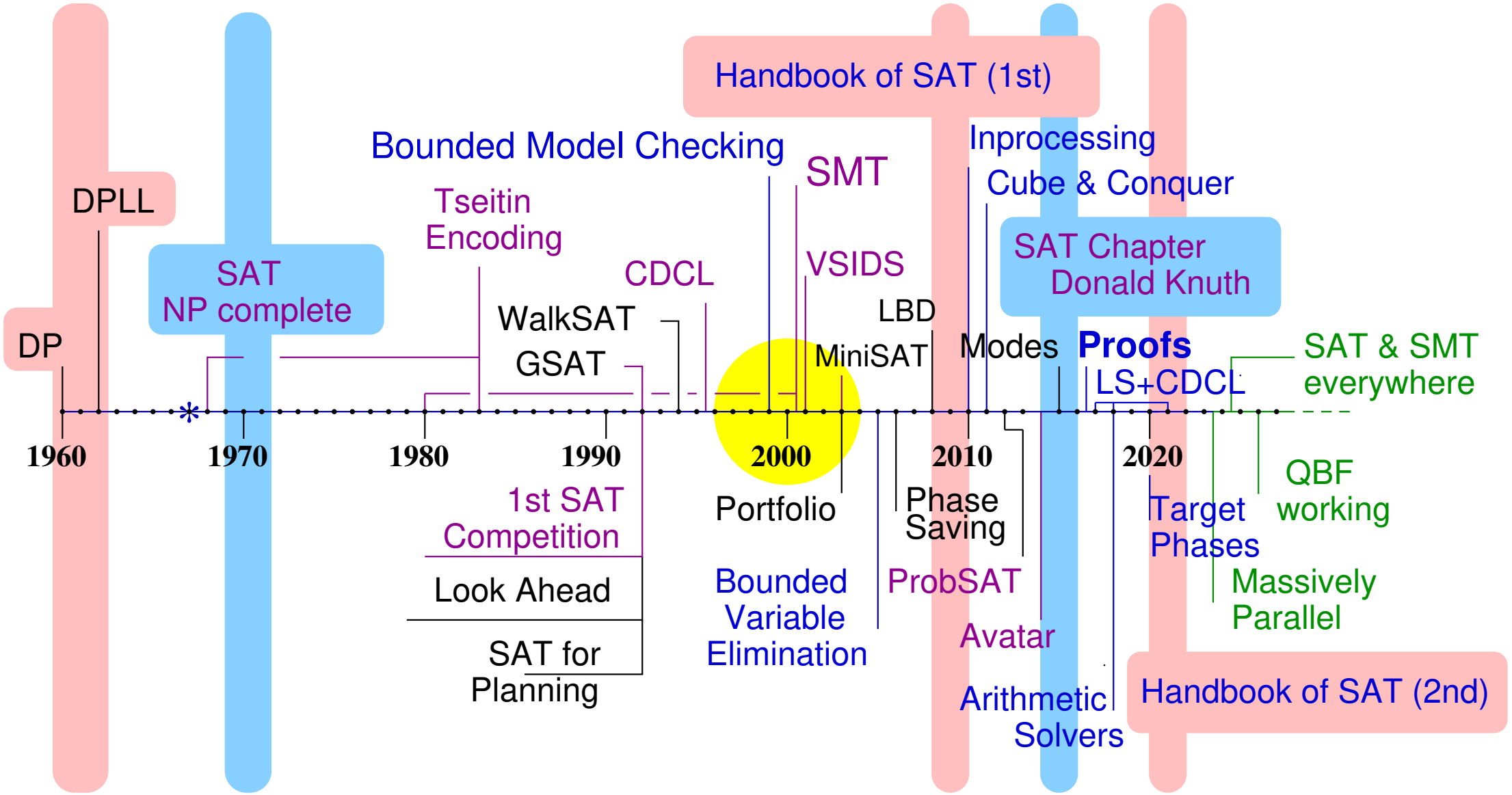
```
generate a random 0/1 initial assignment for all literals L
while (exists unsatisfied clause C in formula) {
  with probability p
    pick literal L in C which breaks minimial number clauses
  otherwise with probability 1-p pick random literal L in C
  flip literal L
}
```

ProbSAT'12

[BalintSchöning-SAT'12]

```
generate a random 0/1 initial assignment for all literals L
while (exists unsatisfied clause C in formula) {
  let s(L) be the number clauses broken by flipping L
  pick L probabilistically with probability 2^-s(L)
  flip literal L
}
```

Satisfiability Modulo Theories (SMT) aka CDCL(T) from around 2000



Satisfiability Modulo Theories (SMT) aka CDCL(T)

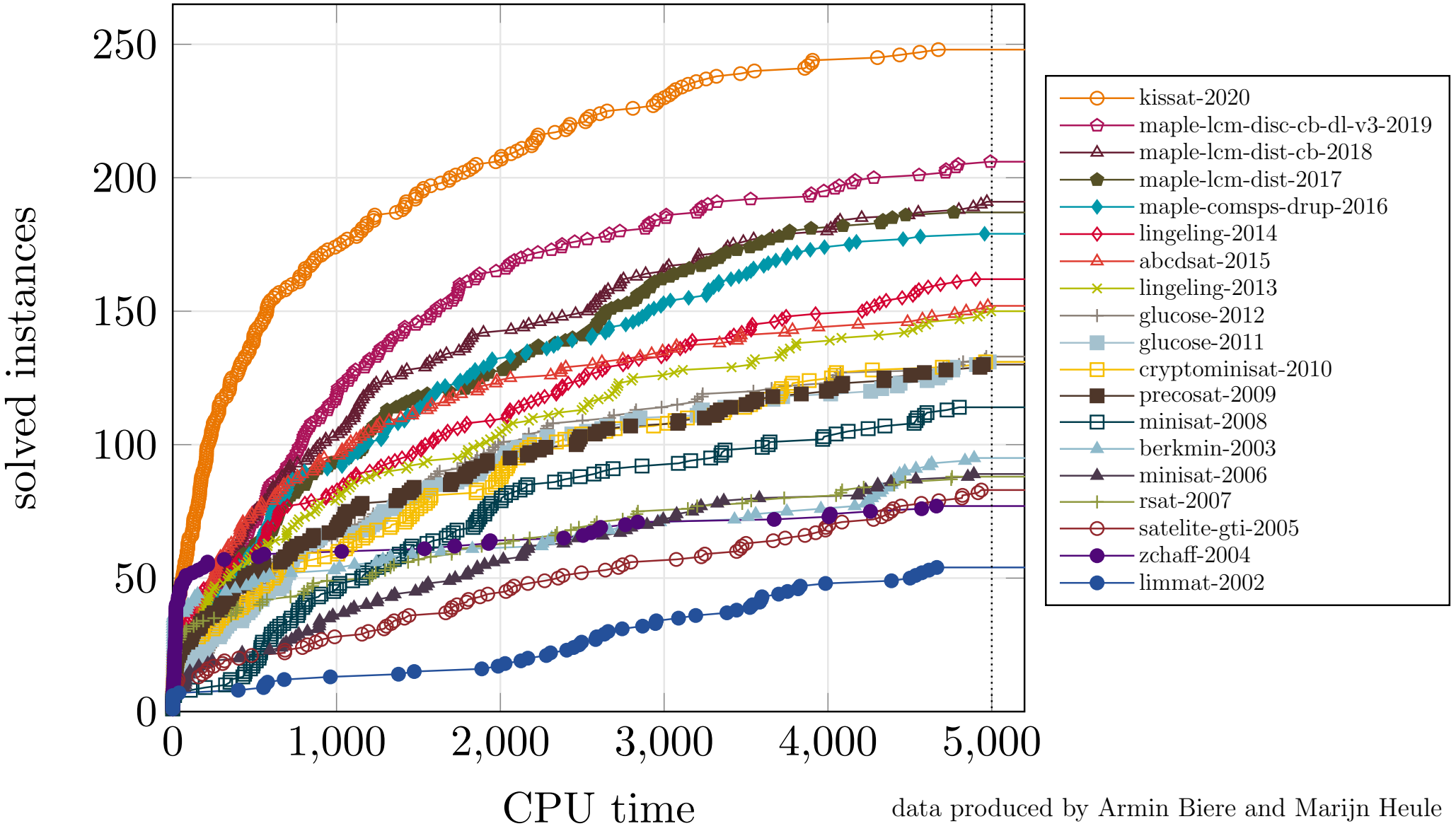
$$f(x) \neq f(y) \;\wedge\; x+u=3 \;\wedge\; v+y=3 \;\wedge\; u=t[z] \;\wedge\; v=t[w] \;\wedge\; z=w$$

- originally unquantified formulas in first-order logic
- interpreted symbols over various theories
  - basic theory of equality, uninterpreted functions
  - arithmetic expressions (linear, non-linear, integer, reals)
  - theory of bit-vectors and floating-points to model bit-precise reasoning of HW/SW
  - arrays (McCarthy axioms) to model memory / pointers
  - strings to model regular expressions
- standardized input-format / semantics SMT-LIB
- popular SMT solvers: CVC, Yices, Z3, Boolector, ...

$$x * 10001000_8 = y \;\wedge\; x \gg 3 = y \;\wedge\; x[3..0] = y[7..4]$$


SAT-Race 2010 Award
"Plingeling"
by Armin Biere
is awarded the title of
Best Parallel Solver
SAT-Race 2010 Award
"Lingeling"
by Armin Biere
is awarded the title of
Second Prize Winn
Tenth International Conference
On Theory and Applications of Satisfiability Testing
Competition 2007
Gold medal
Twelfth International Conference
On Theory and Applications of Satisfiability Testing
Competition 2009
Silver medal
Eighteenth International Conference
on Theory and Applications of Satisfiability Testing
SAT-Race 2015
Main Track
Special Prize:
Most Innovative Solver
Written by:
Armin Biere
3rd Prize
Parallel Track
Best Parallel Solver
2010
SAT-Race
2010
SAT
COMPE
TITION
2013
Winner

SAT Competition Winners on the SC2020 Benchmark Suite

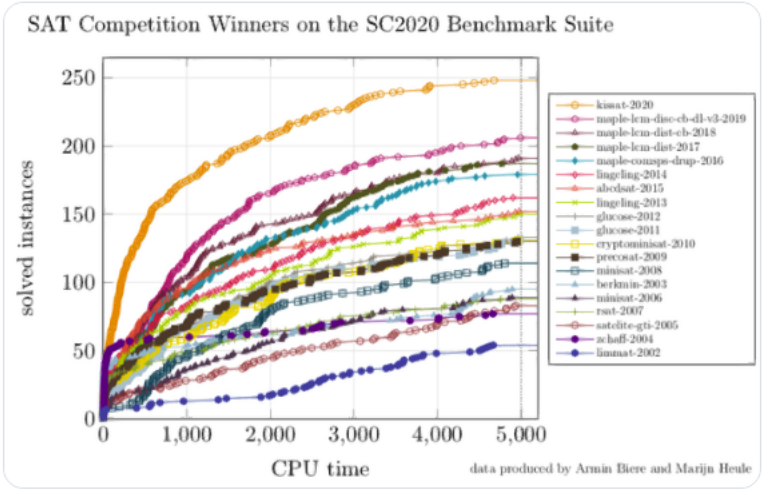


data produced by Armin Biere and Marijn Heule

some Tweets

Armin Biere
@ArminBiere

SAT solvers get faster and faster: all-time winners of the SAT Competition on 2020 instances, featuring our new solver Kissat (fmv.jku.at/kissat), which won in 2020. The web page also has runtime CDFs for 2011 and 2019.

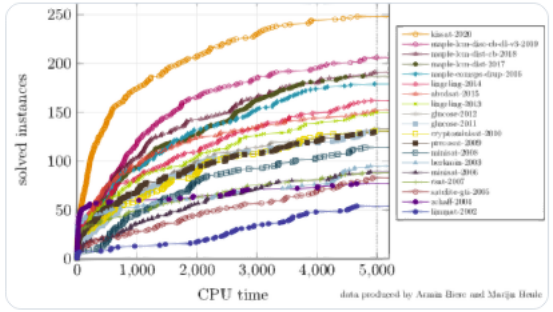


5:20 PM · Jul 28, 2020 · Twitter Web App

View Tweet activity

57 Retweets **7** Quote Tweets **327** Likes

Armin Biere @ArminBiere · Jul 28
SAT solvers get faster and faster: all-time winners of the SAT Competition on 2020 instances, featuring our new solver Kissat (fmv.jku.at/kissat), which won in 2020. The web page also has runtime CDFs for 2011 and 2019.



5 64 327

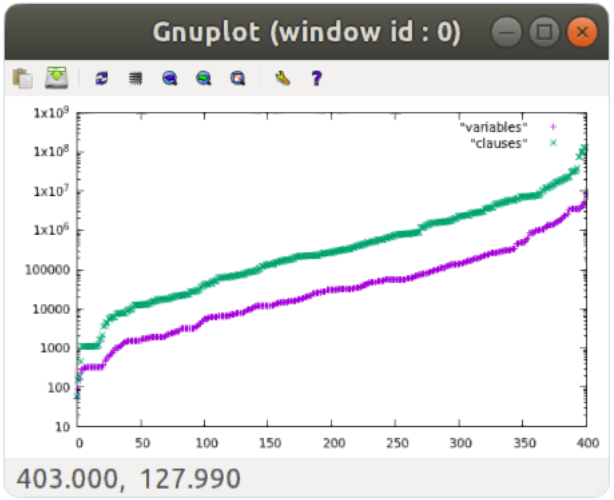
joao @_joaogui1 · Jul 29
How big are the instances?

1 1

Armin Biere
@ArminBiere

Replying to @_joaogui1

The largest ones have millions of variables and clauses. The planning track had even larger ones. See the variable and clause distribution plot for the main track:



Armin Biere
@ArminBiere

Eventually I will need to support 64-bit variable indices (Lingeling has 2^27, CaDiCaL indeed 2^31 and Kissat 2^28 as compromise though it could easily do half a billion)

T-Mobile A 89 % 21:12

Hi,
We are trying to verify Deep Neural Networks with our verification machine ESBMC, that uses Boolector. Our experiments are geting the following error:

- internal error in 'lglib.c': more than 134217724 variables.
  Could we increase this variable number? Since we are performing our experiments in a huge RAM memory.

—
You are receiving this because you are subscribed to this thread.
Reply to this email directly, view it on GitHub, or unsubscribe..

Andrew V. Jones 13:40
an Boolector/boolector, S...

Can you try compiling Boolector with a different SAT solver? I believe that CaDiCaL has a much higher limit (maybe INT_MAX vars).

Zitierten Text anzeigen

Aina Niemetz 18:16
an Boolector/boolector, S...

As @andrewvaughanj points out, this is a limitation in the SAT solver that we can not control. Let me add that CaDiCaL typically outperforms Lingeling in combination with Boolector, so it might be a good idea to switch to CaDiCaL anyways.

more

Tweets

Daniel Le Berre @dleberre · 4 T

This week, we will have again many great talks about SAT at @SimonsInstitute: I am especially looking forward the one of @ArminBiere titled « A Personal History of Practical SAT Solving » simons.berkeley.edu/talks/tbd-308

1 5 19

Daniel Le Berre @dleberre · 4 T

Did you know that @ArminBiere participated to every SAT competitive event since 2002? And won in a fair amount of tracks? See all his solvers here: fmv.jku.at/software/index...

1 3

Daniel Le Berre

@dleberre

I remember the solvers Limmat, Compsat, NanoSAT, PicoSAT, PrecoSAT, Lingeling, Splatz, CaDiCaL, Kissat plus « educational » solvers Cleaneling and Satch. Would love to see a cactus plot with all those solvers on Thursday @ArminBiere :)

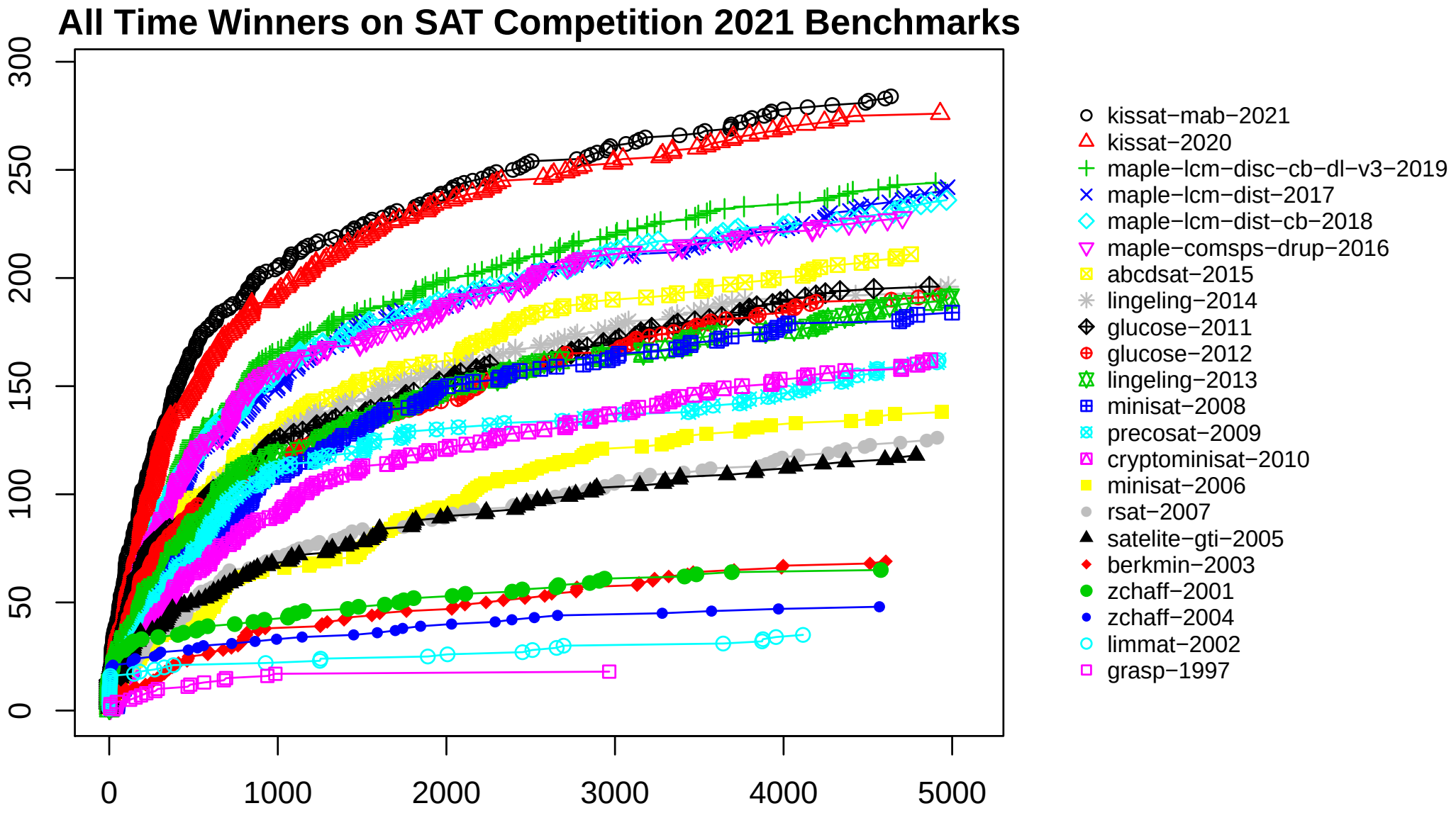
Adam P. Goucher

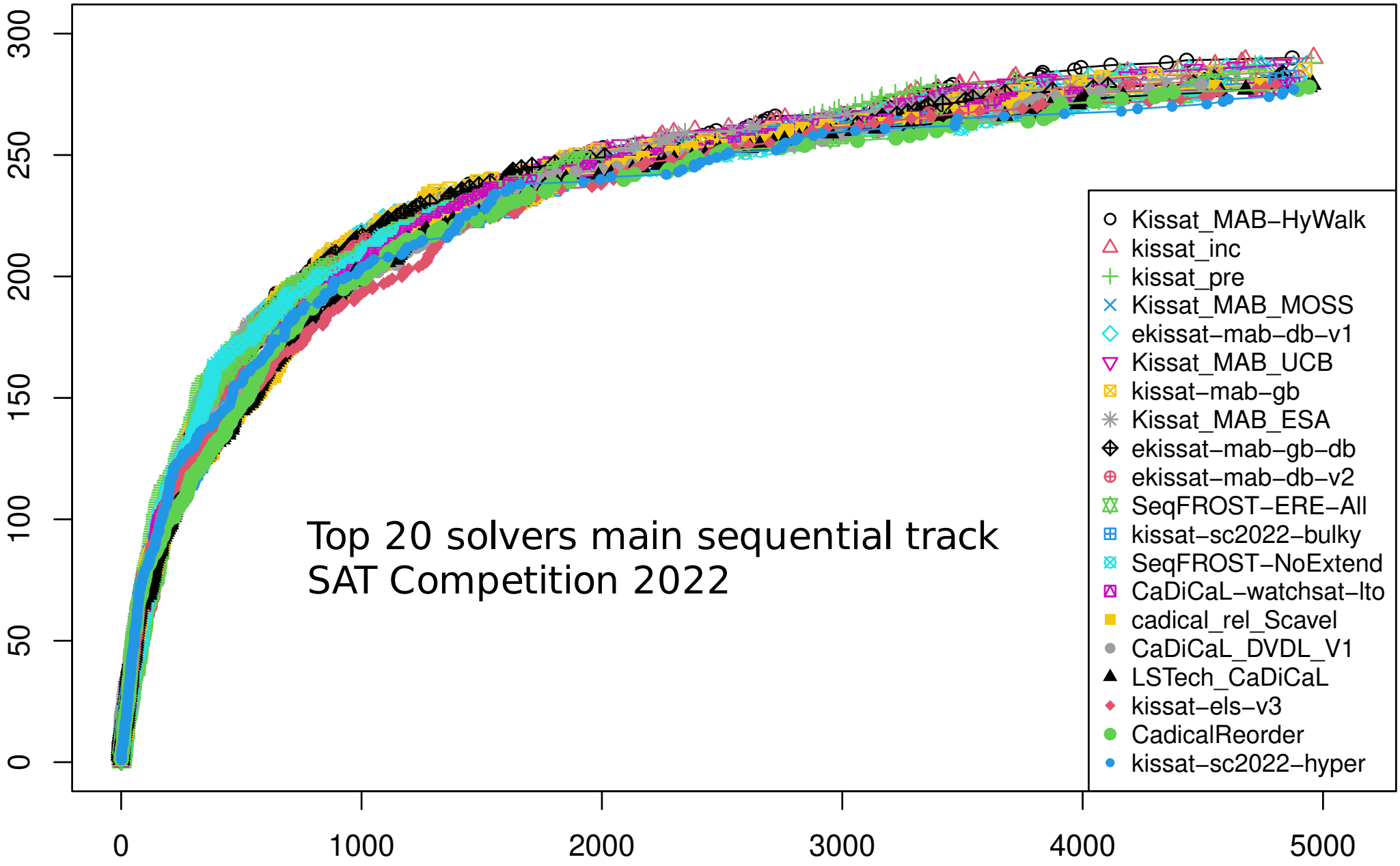
@apgox

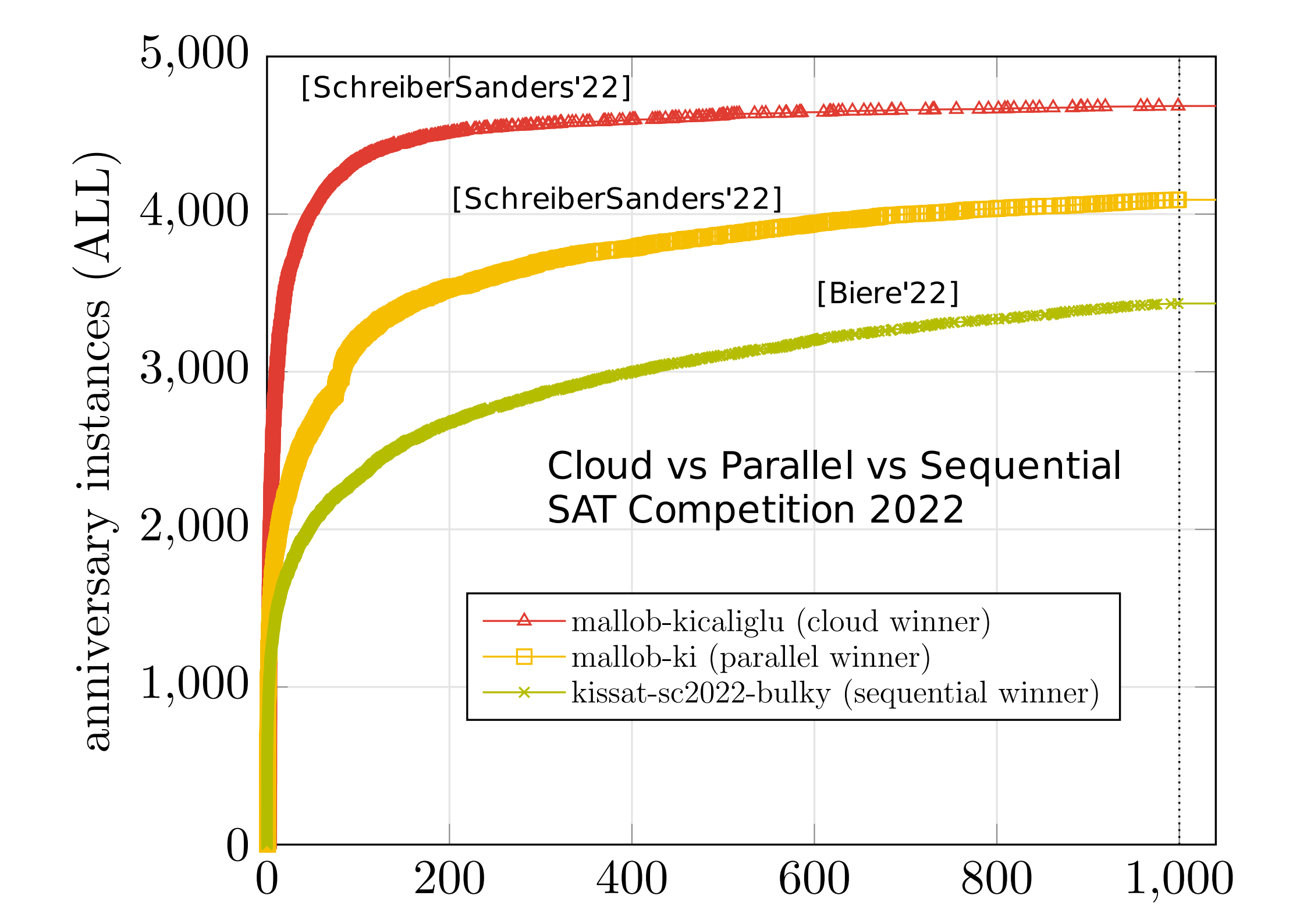
Discovered the first oblique glider rake in @conwaylife (based on a partial discovered by @inspirehep23) by running ikpx2 for 35000 core-hours on the under-utilised CPUs of a @LeaderGPU machine.

This wouldn't have been possible without @ArminBiere's SAT solvers (cadical/kissat).

Tweet übersetzen


All Time Winners on SAT Competition 2021 Benchmarks
0
50
100
150
200
250
300
0
1000
2000
3000
4000
5000
kissat−mab−2021
kissat−2020
maple−lcm−disc−cb−dl−v3−2019
maple−lcm−dist−2017
maple−lcm−dist−cb−2018
maple−comsps−drup−2016
abcdsat−2015
lingeling−2014
glucose−2011
glucose−2012
lingeling−2013
minisat−2008
precosat−2009
cryptominisat−2010
minisat−2006
rsat−2007
satelite−gti−2005
berkmin−2003
zchaff−2001
zchaff−2004
limmat−2002
grasp−1997


Top 20 solvers main sequential track
SAT Competition 2022
0
50
100
150
200
250
300
0
1000
2000
3000
4000
5000
Kissat_MAB−HyWalk
kissat_inc
kissat_pre
Kissat_MAB_MOSS
ekissat−mab−db−v1
Kissat_MAB_UCB
kissat−mab−gb
Kissat_MAB_ESA
ekissat−mab−gb−db
ekissat−mab−db−v2
SeqFROST−ERE−All
kissat−sc2022−bulky
SeqFROST−NoExtend
CaDiCaL−watchsat−lto
cadical_rel_Scavel
CaDiCaL_DVDL_V1
LSTech_CaDiCaL
kissat−els−v3
CadicalReorder
kissat−sc2022−hyper


5,000
4,000
3,000
2,000
1,000
0
anniversary instances (ALL)
0
200
400
600
800
1,000
[SchreiberSanders'22]
[SchreiberSanders'22]
[Biere'22]
Cloud vs Parallel vs Sequential
SAT Competition 2022
mallob-kicaliglu (cloud winner)
mallob-ki (parallel winner)
kissat-sc2022-bulky (sequential winner)

Migrating Solver State [BiereChowdhuryHeuleKieslWhalen'SAT22]

Armin Biere
University of Freiburg, Germany

Md Solimul Chowdhury
Carnegie Mellon University, USA

Marijn J.H. Heule
Carnegie Mellon University,
Amazon Web Services, Inc., USA

Benjamin Kiesl
Amazon Web Services, Inc., Germany

Michael W. Whalen
Amazon Web Services, Inc.,
The University of Minnesota, USA

Abstract

We present approaches to store and restore the state of a SAT solver, allowing us to migrate the state between different compute resources, or even between different solvers. This can be used in many ways, e.g., to improve the fault tolerance of solvers, to schedule SAT problems on a restricted number of cores, or to use dedicated preprocessing tools for inprocessing. We identify a minimum viable subset of the solver state to migrate such that the loss of performance is small. We then present and implement two different approaches to state migration: one approach stores the state at the end of a solver run whereas the other approach stores the state continuously as part of the proof trace. We show that our approaches enable the generation of correct models and valid unsatisfiability proofs. Experimental results confirm that the overhead is reasonable and that in several cases solver performance actually improves.

2012 ACM Subject Classification Theory of computation

Keywords and phrases SAT, SMT, Cloud Computing, Serverless Computing

Tweet

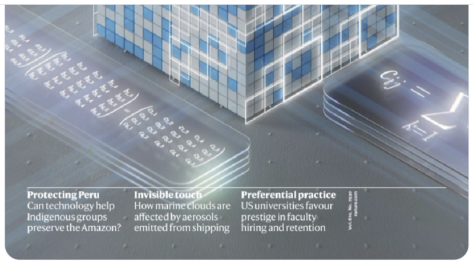
Māris Ozols
@enclanglement

Humanity strikes back: two 5×5 matrices over $\mathbb{Z}_2$ require only 95 instead of 96 multiplications. Take it AlphaTensor!
arxiv.org/abs/2210.04045

> **DeepMind** @DeepMind · 6d
> ICYMI: On the cover of @Nature - #AlphaTensor, an AI system for discovering novel, efficient, and exact algorithms for matrix multiplication.
>
> Learn more ⬇️...

Tweet your reply

Tweet



10:14 · 11 Oct 22 · Twitter Web App

133 Retweets **23** Quote Tweets **761** Likes

Ryan O'Donnell @BooleanAnalysis · 1d
Replying to @enclanglement
Still 'computer-assisted' if I know Manuel Kauers, but yeah, this is pretty neat :) Now we need SAT-solvers to give us better lower bounds...

3 1 23

Māris Ozols @enclanglement · 1d
Yes, the lines are certainly blurry here. This reminds me of "centaur chess".

Advanced chess - Wikipedia
en.wikipedia.org

Tweet your reply

Kronen Zeitung

Teilen 0

BUNDESLÄNDER > OBERÖSTERREICH
18.10.2022 15:30

MATHE-GENIES

Mit einem Schritt zum neuen Weltrekord

Tüftler: Jakob Moosbauer und Manuel Kauers (re.) (Bild: Dostal Harald)

Den Mathematikern Manuel Kauers und Jakob Moosbauer von der JKU ist es gelungen, den erst kürzlich aufgestellten Rekord für 5x5-Matrizen zu toppen. Sie konnten die schwierige Rechenoperation um einen Schritt verkürzen und freuen sich nun über den Weltrekord. Die beiden rechnen zudem schneller als Künstliche Intelligenz.

DERSTANDARD

Wissenschaft International Inland Wirtschaft

WAS IST DIE MATRIX?

Linzer Mathematiker übertrumpfen künstliche Intelligenz

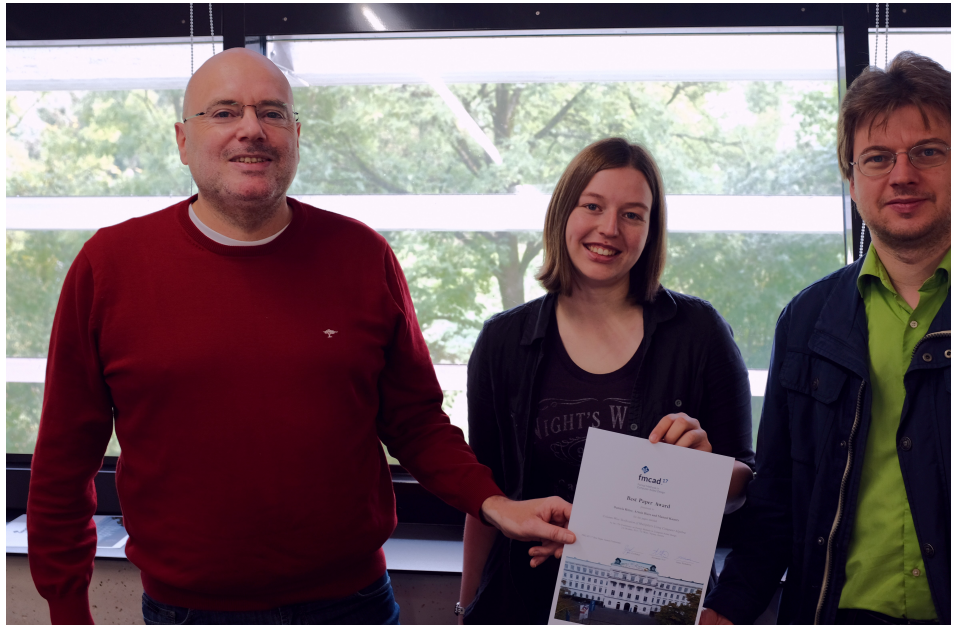
Forscher der Johannes-Kepler-Universität fanden eine neue Methode, wie man besonders effizient 5x5-Matrizen multipliziert. Sie stechen dabei Computermethoden aus

Reinhard Kleindl
19. Oktober 2022, 14:10, 22 Postings

Was ist einfach, und was ist schwierig? Die Beantwortung dieser Frage gehört selbst zu den schwierigen Fragen. (Was wiederum die Frage aufwirft, ob sich Letzteres so einfach sagen lässt.) Darum geht es auch in der Mathematik, auf dem Gebiet der "Berechenbarkeit". Hier wird den technischsten und abstraktesten

Machine Learning || Automated Reasoning

Arithmetic Solvers



OESTERREICHISCHE COMPUTER GESELLSCHAFT
AUSTRIAN COMPUTER SOCIETY

GESELLSCHAFT FÜR INFORMATIK

Die Gesellschaft für Informatik e.V. (GI), die Oesterreichische Computer Gesellschaft (OCG) sowie die Schweizer Informatikgesellschaft (SI) verleihen

Frau Dr. Daniela Kaufmann
für ihre hervorragende Dissertation
„Formal Verification of Multiplier Circuits using Computer Algebra“
den GI-Dissertationspreis 2020.

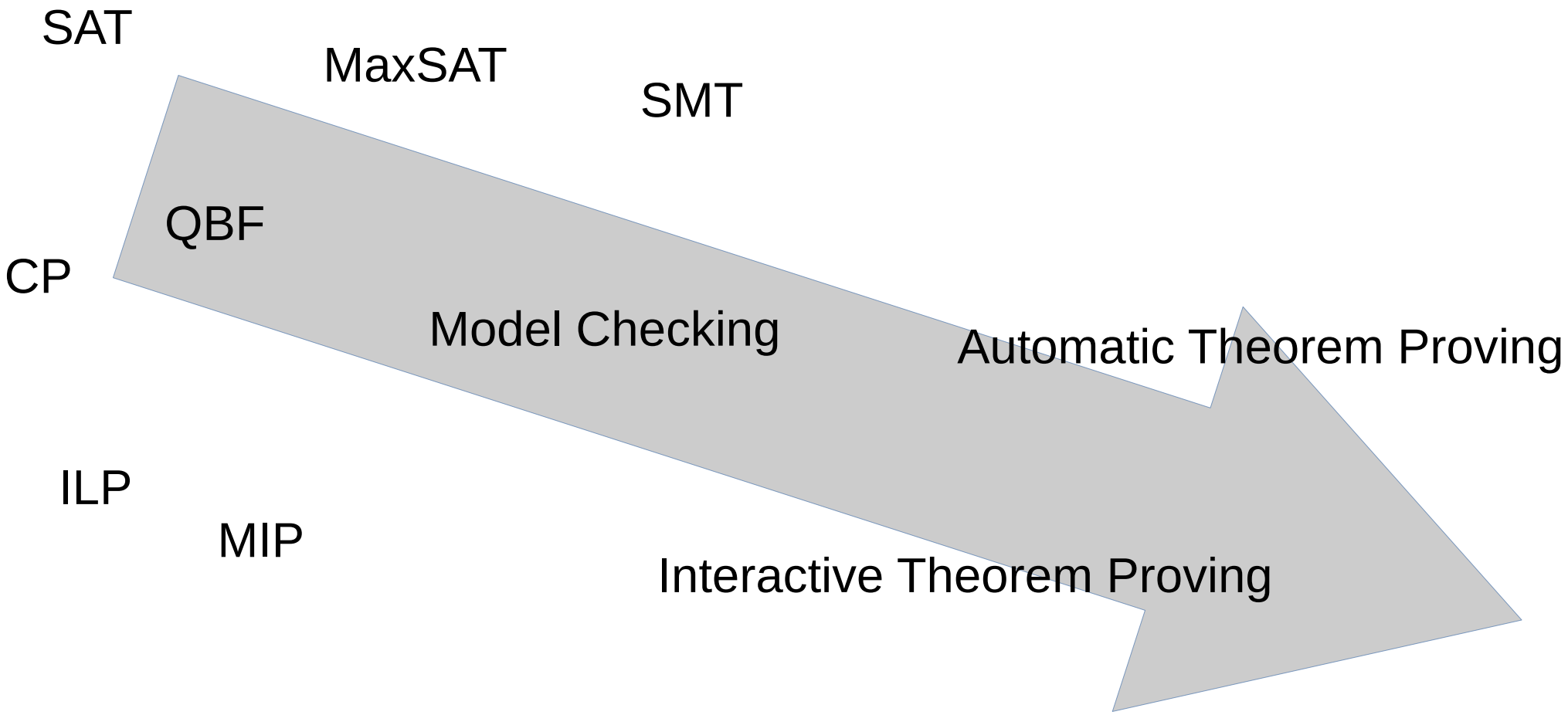
Frau Dr. Kaufmann hat aktuelle Verifikationsmethoden basierend auf Computeralgebra verbessert und neue Methoden entwickelt, die für einen gegebenen Integer-Multiplizierer auf Gatterebene vollautomatisch über dessen Korrektheit entscheiden, ohne dass die Entwicklerinnen und Entwickler manuell in den Verifikationsprozess eingreifen müssen.

Mit dieser Preisverleihung würdigen die beteiligten Gesellschaften – die Gesellschaft für Informatik e.V. (GI), die Schweizer Informatik Gesellschaft (SI) und die Osterreichische Computergesellschaft (OCG) – eine herausragende Arbeit, die eine überraschende Lösung für ein lange bekanntes Problem im Spektrum Theorie, Software und industrielle Anwendung liefert.

Berlin, im September 2021

Prof. Dr. Hannes Federrath
Präsident der Gesellschaft für Informatik e.V. (GI)

Automated Reasoning Technologies



Challenges

- Local Search for UNSAT
- Parallel Automated Reasoning
- Even more Scalability and Automation
- Combining Precise and Inprecise Reasoning
- Education in Logic and Automated Reasoning