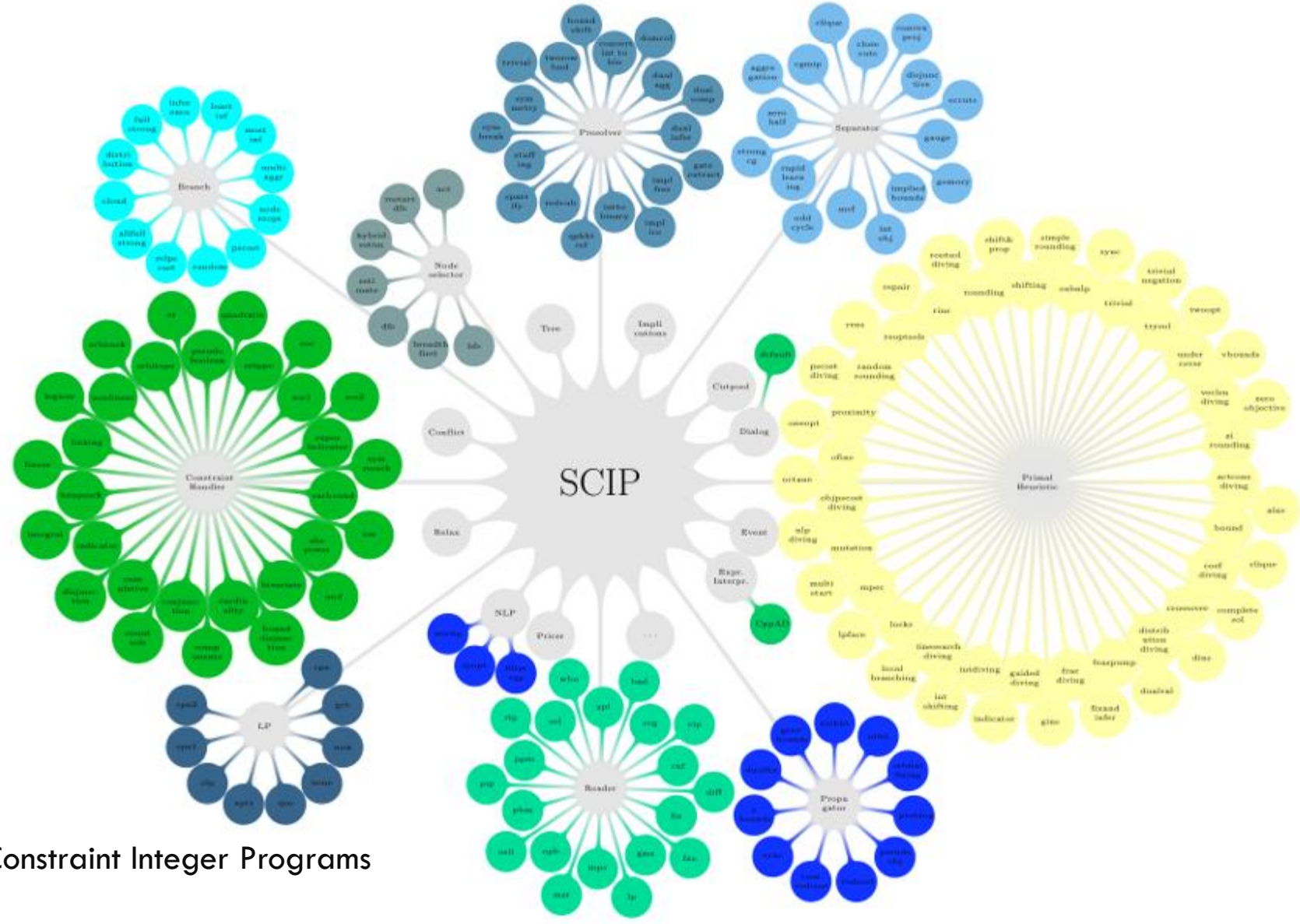
SCIP ORIGINS



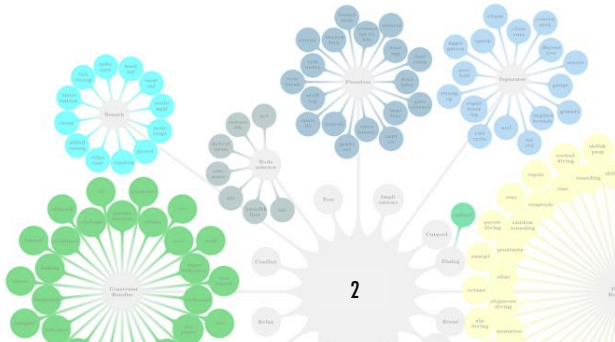
Let's SCIP it!
Workshop to celebrate 20 years of Solving Constraint Integer Programs

November 4, 2022

Tobias Achterberg

HOW IT BEGAN

```
commit 147d536831454c6997f7c5352cc1e4711b7d0f88
Author: Tobias Achterberg <achterberg@zib.de>
Date:   Wed Oct 23 14:31:34 2002 +0000

    - initial directory structure and configuration files
```

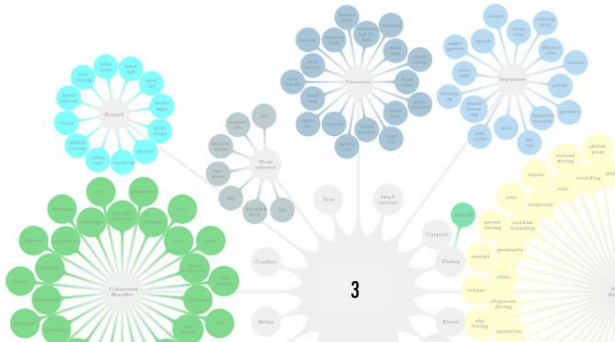


HOW IT REALLY BEGAN

```
$ cd ~/projects/scip
$ cvs import –m "- initial directory structure and
configuration files" scip zib start
```

The switch from cvs to git happened much later...

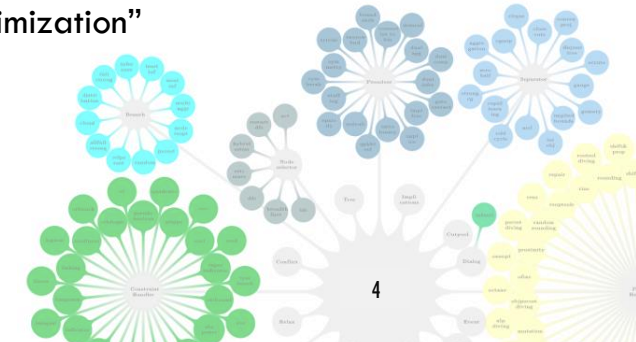
```
commit b03e967a66b05236ae137e4e32d2f9357c20c5b5
Author: Matthias Miltenberger <miltenberger@zib.de>
Date:   Mon Mar 7 10:50:13 2011 +0100

    converted .cvsignore files to .gitignore
```



PRECURSORS

Studied math and computer science at TU Berlin

- Implemented textbook simplex algorithm in Martin Grötschel's "Linear Optimization" class
  - Most difficult piece: LP file format parser
  - Together with Utz-Uwe Haus, who knew yacc/bison
- Joined Zuse Institute Berlin (ZIB) in 1999
- Math: discrete optimization (Prof. Martin Grötschel)
  - Master's thesis (2000): "Arboreszenz-Flüsse in Graphen: polyedrische Untersuchungen"
- CS: artificial intelligence (Prof. Klaus Obermayer)
  - Master's thesis (2001): "Anwendung von neuronalen Netzen zur Beschleunigung von Branch & Bound Verfahren der Kombinatorischen Optimierung"
    - English: "Applications of neural networks for speeding up branch-and-bound methods in combinatorial optimization"
    - Needed MIP code!



SIP

Thorsten Koch

Alexander Martin

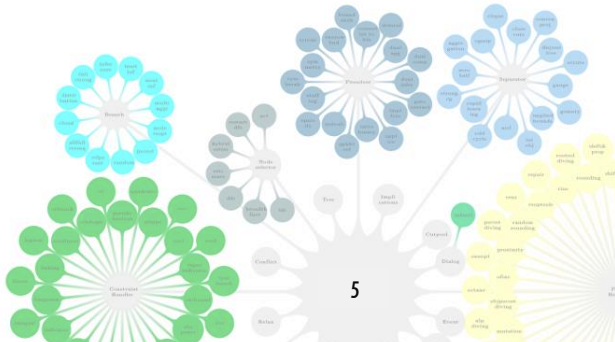
Solving Integer Programs

- MIP code implemented by Alexander Martin in 1998
  - Used CPLEX as LP solver
  - Help from Bob Bixby to make best use of CPLEX
    - Including hidden “FastMIP” setting and strong branching
  - Had state-of-the-art components
    - Presolving
    - Parallel tree search
    - Cutting planes (Gomory, MIR, knapsack, ...)
    - Pseudo-cost branching with strong branching initialization
  - Comparable in speed to then-current CPLEX 6.0

Alexander Martin moved to TU Darmstadt in 2000

Thorsten Koch introduced me to SIP

- Thankfully, Alex allowed me to use his code for my research

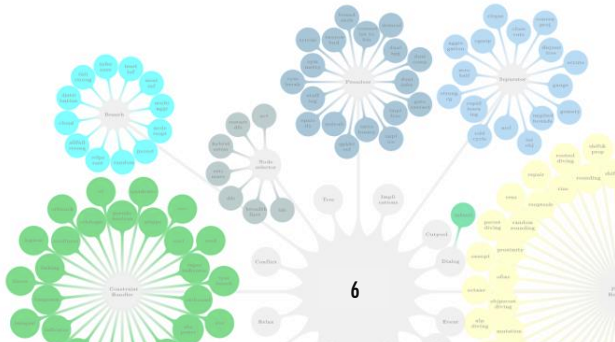


MACHINE LEARNING FOR BRANCHING RULES

Needed a reason to do optimization for my CS master's thesis

Proposed to use AI methods to learn a good branching rule in SIP

- Goal:
  - Given a feature vector for a specific branching variable candidate
  - Estimate the objective gains for the downwards and upwards branch
- Consider various features
  - Column length, variable class, fractionality, pseudo costs, history value, distance to root LP solution, ...
  - Collect data: feature vectors and objective gains
- Train machine learning model to map feature vectors to objective gains
  - Principal component analysis
  - Multi-layer perceptron (aka neural network)
  - Autoencoder
  - Radial basis functions (with k-means clustering)
  - Support vector machines
  - Mixtures of experts



MACHINE LEARNING FOR BRANCHING RULES

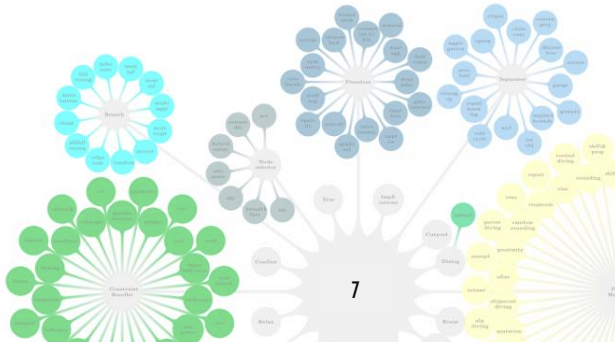
Needed a reason to do optimization for my CS master's thesis

Proposed to use AI methods to learn a good branching rule in SIP

- Goal:
  - Given a feature vector for a specific branching variable candidate
  - Estimate the objective gains for the downwards and upwards branch
- Consider various features
  - Column length, variable class, fractionality, pseudo costs, history value, distance to root LP solution, ...
  - Collect data: feature vectors and objective gains
- Train machine learning model to map feature vectors to objective gains
  - Principal component analysis
  - Multi-layer perceptron (aka neural network)
  - Autoencoder
  - Radial basis functions (with k-means clustering)
  - Support vector machines
  - Mixtures of experts

Result:
History values are good predictors

⇒ Reliability Branching



WHAT NEXT?

MIP development is super cool!

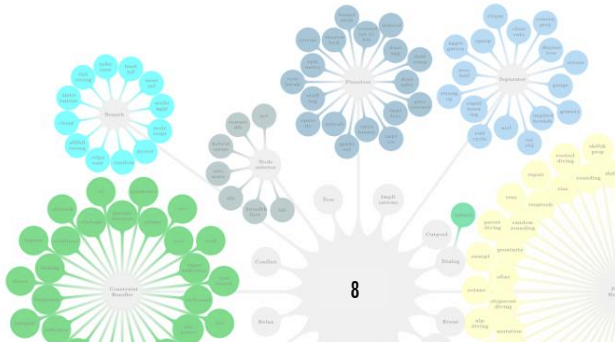
Learned in my CS classes that there are other, similar techniques

- Logic programming
- SAT solving
- Constraint programming

Pretty much the same as MIP, just without LP relaxation!

Idea: combine all of this

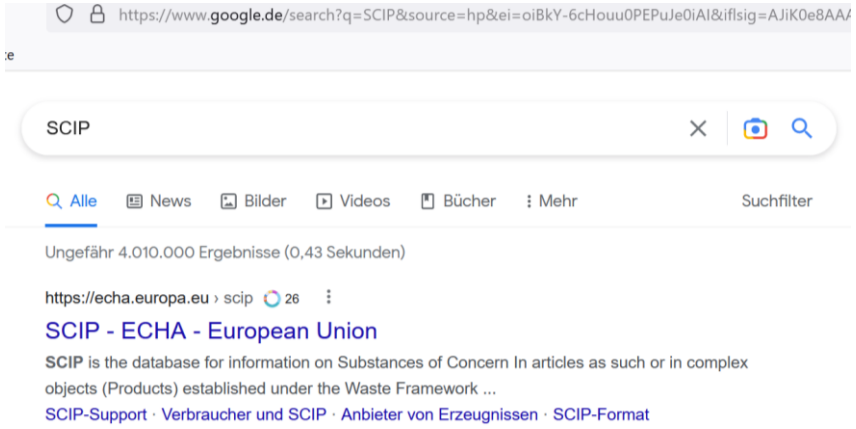
- Realized that this is impossible in data structures of SIP
- Thus, started to write a new solver from scratch
- Really bad at inventing names
  - So I just copied Alexander Martin's name and added a "C"
  - It was a sub-optimal choice...

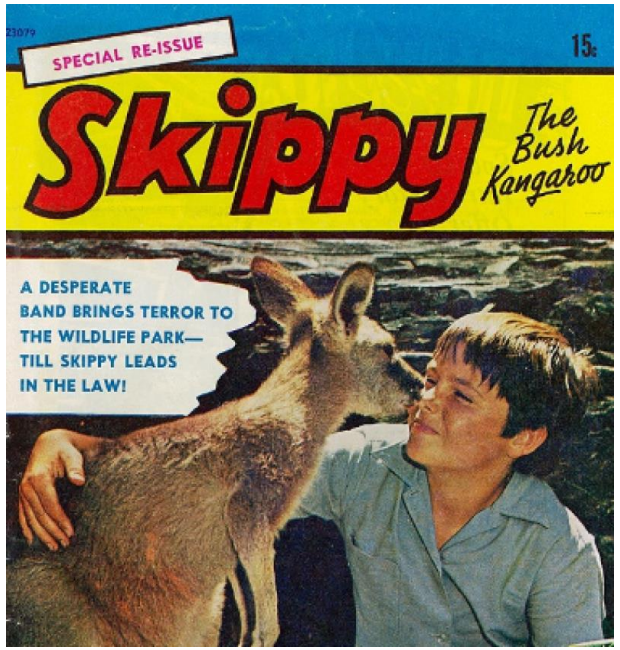


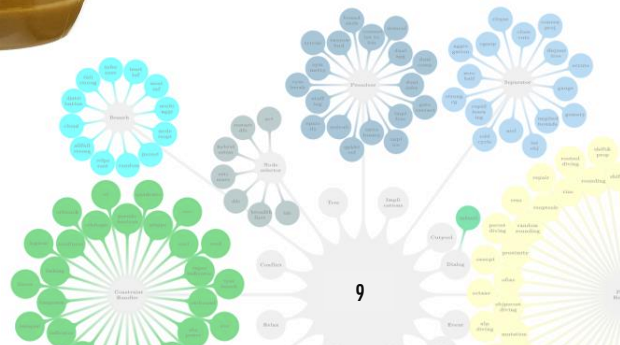
SCIP

What is SCIP?

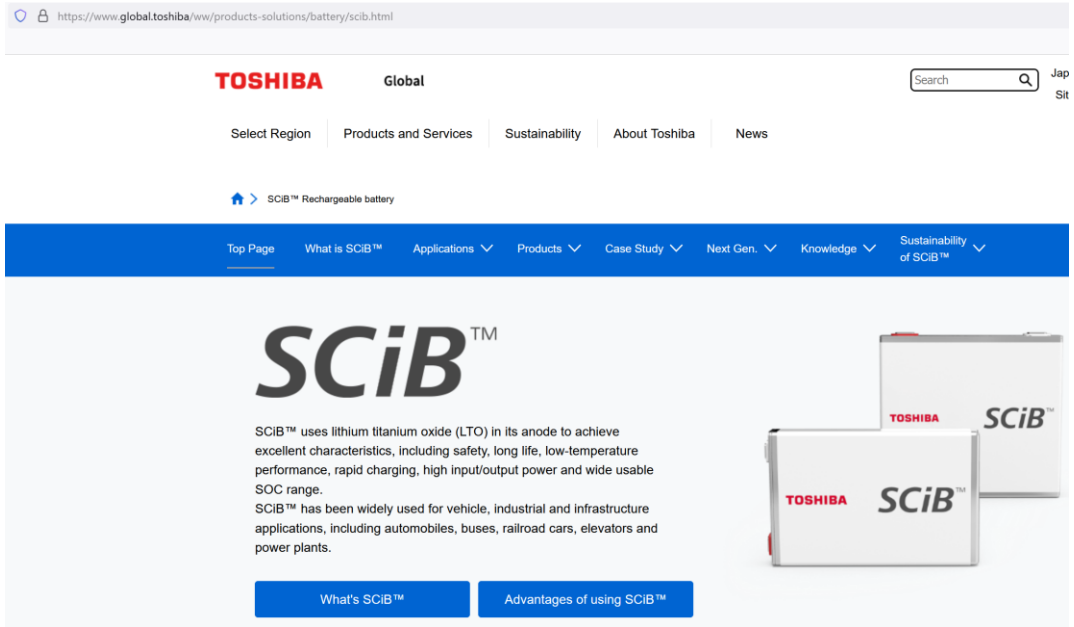
Specifications Consultants in Independent Practice is an international technical resource organization which assists design firms, owners, and manufacturers in acquiring professionally written construction specifications from qualified independent and employed specifiers, who advance excellence in preparing construction specifications, promote their special services and expertise to potential clients, share their knowledge, experience, and resources through discussions, conferences, and educational programs, network for mutual benefit.

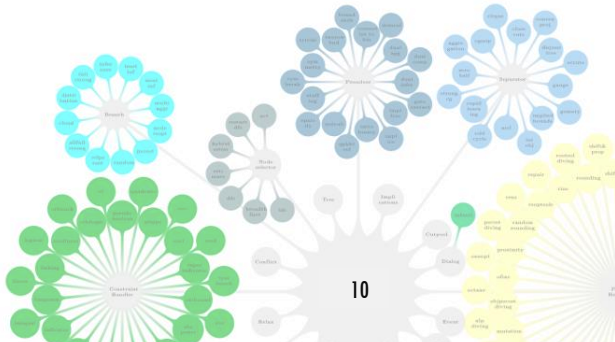






REGISTER SCIP AS A TRADEMARK





BACK TO CONSTRAINTS

What is MIP?

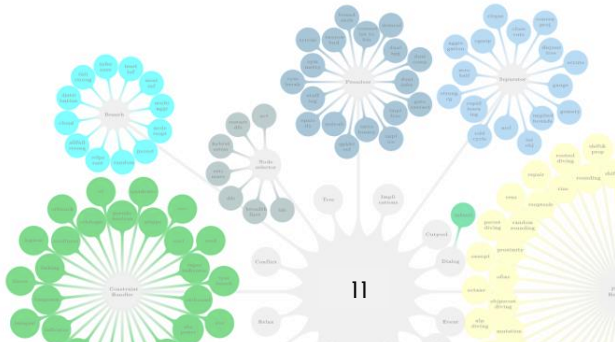
- MIP = CP for linear constraints + LP relaxation

What can you do algorithmically with constraints?

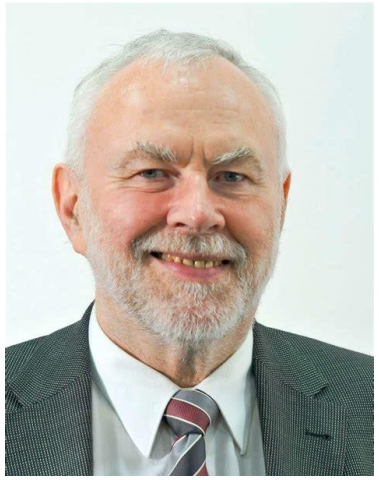
- Presolving
- Propagation
- LP relaxation
- Cutting plane separation
- Solution checking
- Branching

Plug-in concept for constraints

- Sounds like C++, right?
- But back in 2002, C++ was a mess (and still is)
- C code is fast, portable, and can use a powerful C preprocessor



AND WHY?



Martin Grötschel

Raik Brinkmann

Extraordinary luck

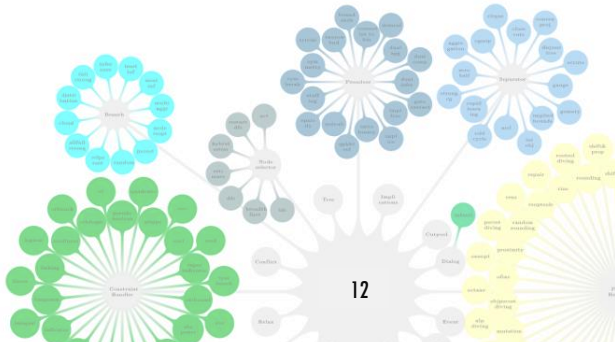
- Prof. Grötschel had contact to Infineon in Munich
- They had problems with verifying chip designs with arithmetic circuits
- Ideal for combining MIP, CP, and SAT
  - MIP and LP are good in dealing with arithmetics
  - CP and SAT are good in dealing with logic
- Infineon's technology was based on SAT

Later, Infineon's chip design group was spun off

- OneSpin solutions
- Acquired by Siemens in 2021

What is the biggest challenge in applied projects?

- Data is missing
- Data comes way too late
- Data has errors
- Not so with Infineon!
  - They had lots of data sets, and data was very clean!



GETTING INTO SAT SOLVING

One day, Prof. Grötschel asked me to pick up Yakov Novikov from Berlin Zoologischer Garten

- Yakov is a SAT researcher from Minsk (Belarus)
  - BerkMin (with E. Goldberg from Berkley)
- Needed to go to Infineon/Munich, passed by in Berlin to visit ZIB

Yakov Novikov

ELSEVIER

Discrete Applied Mathematics
Volume 155, Issue 12, 15 June 2007, Pages 1549-1561

BerkMin: A fast and robust Sat-solver

Eugene Goldberg [a], Yakov Novikov [b]

Show more

+ Add to Mendeley Share Cite

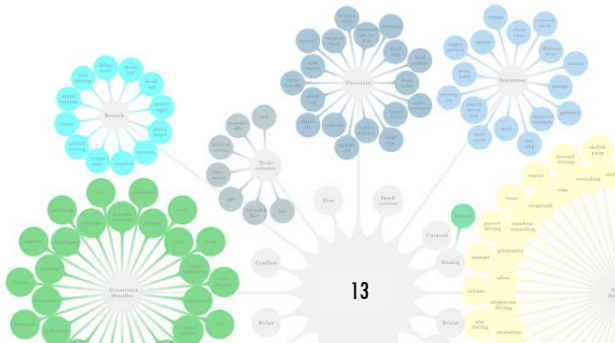
https://doi.org/10.1016/j.dam.2006.10.007

Get rights and content

Open archive

Abstract

We describe a SAT-solver, BerkMin, that inherits such features of GRASP, SATO, and Chaff as clause recording, fast BCP, restarts, and conflict clause "aging". At the same time BerkMin introduces a new decision-making procedure and a new method of clause database management. We experimentally compare BerkMin with Chaff, the leader among resolution-based SAT-solvers. Experiments show that our program is more robust than Chaff being able to solve more instances than Chaff in a reasonable amount of time.



WHAT YOU CAN LEARN IN THE SUBWAY

Yakov Novikov

Trip from Zoologischer Garten to ZIB

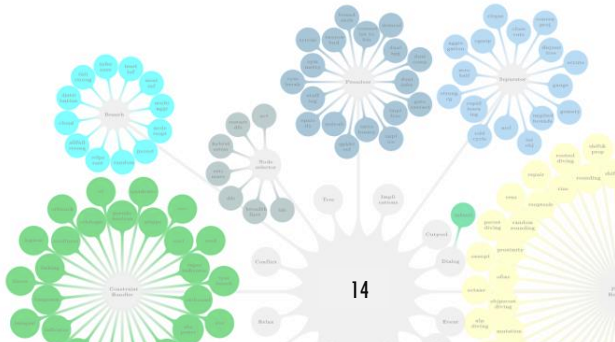
- Take subway U9 from Zoo to Rathaus Steglitz (12 minutes)
- Then bus X83 from Rathaus Steglitz to Arnimallee (6 minutes)

Yakov explained the three important performance ingredients of SAT solvers

- Two-watched literals scheme
  - To perform domain propagation of SAT clauses
- Conflict analysis
  - To learn from infeasible sub-problems
- Restarts

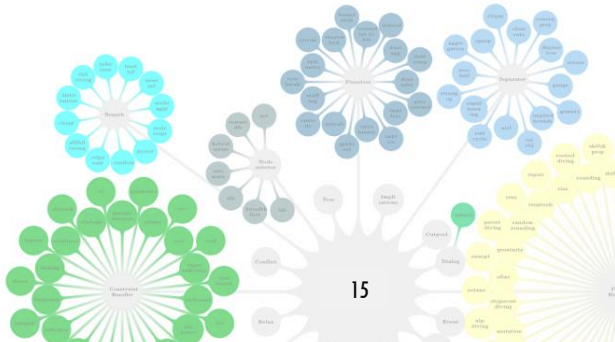
Soon after, I implemented all of them in SCIP

- Two-watched literals scheme for set covering constraint propagation
- Generalized conflict analysis to MIP for infeasible LP relaxations
- Restarts at root node (tree restarts didn't help performance in SCIP)



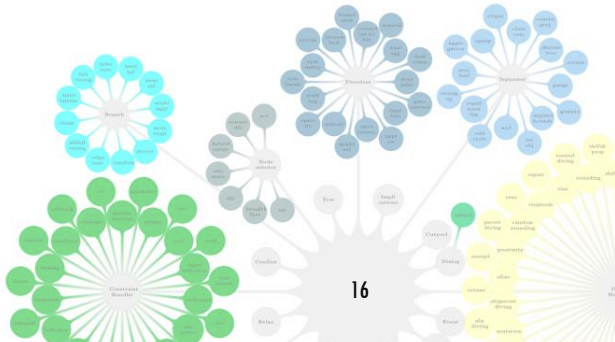
YOU NEVER WALK ALONE...

```
0001 (2002-10-23): Tobias Achterberg (achterberg@zib.de)
0002 (2004-01-15): Marc Pfetsch (m.pfetsch@tu-bs.de)
0003 (2004-03-09): Kati Wolter (wolter@zib.de)
0004 (2005-01-31): Timo Berthold (berthold@zib.de)
0005 (2005-09-13): Annegret Dix (dix@zib.de)
0006 (2006-05-10): Benjamin Hiller (hiller@zib.de)
0007 (2006-05-22): Stefan Heinz (heinz@zib.de)
0008 (2006-09-03): Sebastion Orlowski (orlowski@zib.de)
0009 (2007-01-15): Marika Neumann (marika.neumann@zib.de)
0010 (2007-09-21): Lingfeng Niu (lingfeng@zib.de)
0011 (2007-09-24): Thorsten Koch (koch@zib.de)
0012 (2008-02-14): Michael Winkler (michael.winkler@zib.de)
0013 (2008-02-26): Robert Waniek (waniek@zib.de)
0014 (2008-06-05): Gerald Gamrath (gamrath@zib.de)
0015 (2008-07-09): Christian Raack (raack@zib.de)
0016 (2008-07-24): Stefan Vigerske (vigerske@zib.de)
0017 (2008-09-26): Ambros Gleixner (gleixner@zib.de)
0018 (2009-05-15): Yuji Shinano (shinano@zib.de)
```



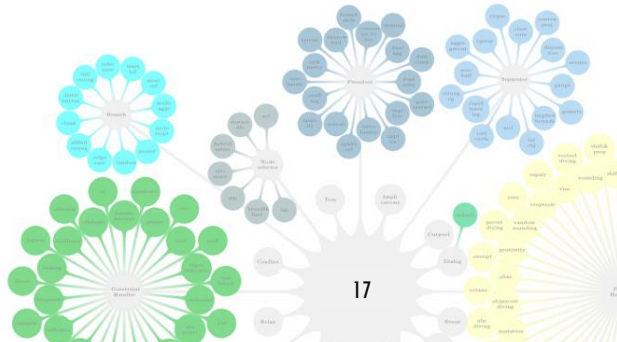
YOU NEVER WALK ALONE...

```
0019 (2009-07-06): Pietro Belotti (belotti@zib.de)
0020 (2009-08-07): Daniel Steffy (steffy@zib.de)
0021 (2009-12-04): Gregor Hendel (hendel@zib.de)
0022 (2010-03-31): Thomas Lehmann (lehmann@zib.de)
0023 (2010-05-27): Andre Augusto Cire (cire@zib.de)
0024 (2010-09-15): Robert Schwarz (schwarz@zib.de)
0025 (2011-03-07): Matthias Miltenberger (miltenberger@zib.de)
0026 (2010-11-26): Humpola (bzfhumpo@opt85.zib.de)
0027 (2011-03-13): pfetsch (m.pfetsch@tu-bs.de)
0028 (2011-07-15): Dan Steffy (steffy@zib.de)
0029 (2011-11-02): Stefan Weltge (weltge@zib.de)
0030 (2012-02-01): Dieter Weninger (dieter.weninger@math.uni-erlangen.de)
0031 (2012-04-10): Frederic Pythoud (pythoud@zib.de)
0032 (2012-06-20): Domenico Salvagnin (dominiqs@gmail.com)
0033 (2012-07-31): Felipe Serrano (serrano@zib.de)
0034 (2013-03-22): Alexandra Kraft (alexandra.kraft@zib.de)
0035 (2013-08-02): Ingmar Vierhaus (vierhaus@zib.de)
0036 (2013-08-08): Jakob Witzig (witzig@zib.de)
```



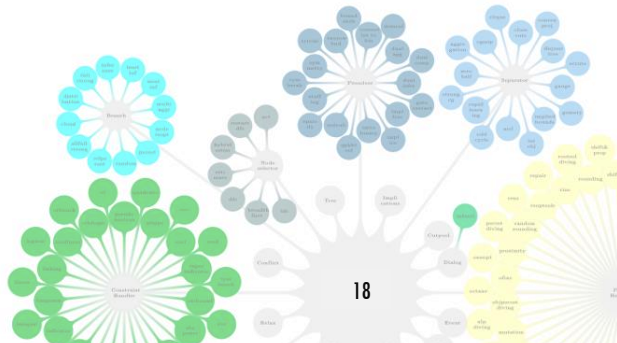
YOU NEVER WALK ALONE...

```
0037 (2013-09-26): Daniel Rehfeldt (rehfeldt@zib.de)
0038 (2013-10-21): Tobias Buchwald (buchwald@zib.de)
0039 (2013-11-11): Benjamin Müller (benjamin.mueller@zib.de)
0040 (2013-12-04): Felix Simon (simon@zib.de)
0041 (2014-01-27): Leif Naundorf (naundorf@zib.de)
0042 (2014-02-12): Timo Strunk (strunk@zib.de)
0043 (2014-02-21): Stephen J Maher (maher@zib.de)
0044 (2014-11-26): Benjamin Mueller (benjamin.mueller@zib.de)
0045 (2015-04-16): Tobias Fischer (fischer@gsc.tu-darmstadt.de)
0046 (2015-10-14): Nagel (bzfnagel@opt64.zib.de)
0047 (2015-10-30): Thomas (nagel@zib.de)
0048 (2016-02-25): Sebastian Schenker (schenker@zib.de)
0049 (2016-02-26): Leon Eifler (eifler@zib.de)
0050 (2016-04-07): Cristina Munoz (munoz@zib.de)
0051 (2016-05-09): Sebastian (schenker@zib.de)
0052 (2016-05-13): Christoph Schubert (schubert@zib.de)
0053 (2016-05-13): Thomas Schlechte (schlechte@zib.de)
0054 (2016-05-27): Matthias Walter (matthias@matthiaswalter.org)
```



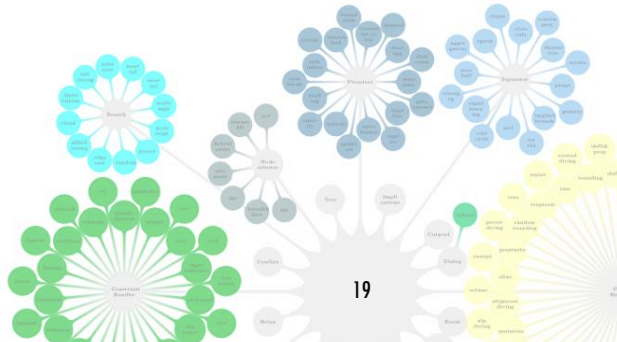
YOU NEVER WALK ALONE...

```
0055 (2016-09-22): Robert Lion Gottwald (robert.gottwald@zib.de)
0056 (2016-10-07): Robert Gottwald (robert.gottwald@zib.de)
0057 (2016-10-04): Tristan Gally (gally@mathematik.tu-darmstadt.de)
0058 (2017-01-31): Renke Kuhlmann (renke.kuhlmann@math.uni-bremen.de)
0059 (2017-03-24): Stephen J. Maher (s.maher3@lancaster.ac.uk)
0060 (2017-05-16): Franziska Schlösser (schloesser@zib.de)
0061 (2017-05-16): Franziska Schloesser (schloesser@zib.de)
0062 (2017-05-25): Pierre Le Bodic (pierre.lebodic@monash.edu)
0063 (2017-06-19): franziska (bzfschlo@optmac01.zib.de)
0064 (2017-07-11): Merlin Viernickel (viernickel@zib.de)
0065 (2017-09-04): Christopher Hojny (hojny@mathematik.tu-darmstadt.de)
0066 (2017-09-15): Fabian Wegscheider (wegscheider@zib.de)
0067 (2017-10-23): pgem (patrick.gemander@fau.de)
0068 (2017-11-24): Patrick Gemander (patrick.gemander@fau.de)
0069 (2018-02-02): Daniel Anderson (daniel.anderson@monash.edu)
0070 (2018-02-20): weninger (dieter.weninger@fau.de)
0071 (2018-04-24): Meskini (bzfmeski@opt35.zib.de)
0072 (2018-05-03): Antonia Chmiela (chmiela@zib.de)
```



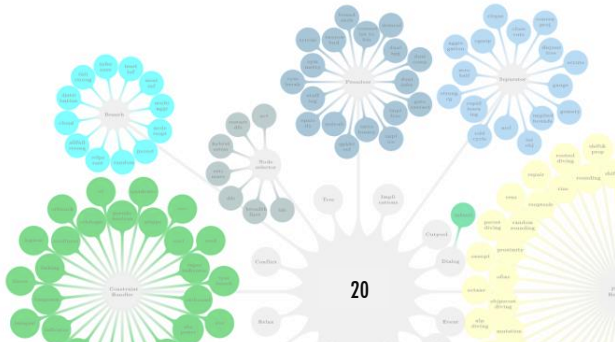
YOU NEVER WALK ALONE...

```
0073 (2018-05-30): Jonas Witt (jonas.witt@rwth-aachen.de)
0074 (2018-06-13): Koichi Fujii (fujii@zib.de)
0075 (2018-07-10): Chen (bzfchenw@opt30.zib.de)
0076 (2018-10-02): Ksenia Bestuzheva (bestuzheva@zib.de)
0077 (2018-10-08): bzfkseni (bestuzheva@zib.de)
0078 (2019-05-14): Helena Mueller (he-mueller-lena@gmx.de)
0079 (2019-06-20): Gabriel Kressin (kressin@zib.de)
0080 (2019-07-18): Katrin Halbig (katrin.halbig@fau.de)
0081 (2019-07-26): Christine Tawfik (tawfik@zib.de)
0082 (2019-08-16): Leona Gottwald (gottwald@zib.de)
0083 (2019-08-29): Sven Fleischer (fleischer@zib.de)
0084 (2019-09-24): Maxime Gasse (maxime.gasse@gmail.com)
0085 (2019-11-06): Frédéric Didier (fdid@google.com)
0086 (2019-11-13): Admin Timo (timo-admin@zib.de)
0087 (2019-11-17): Weikun (cwk@lsec.cc.ac.cn)
0088 (2019-11-18): weikun (cwk@lsec.cc.ac.cn)
0089 (2019-11-21): bzfmulac (mulackal@zib.de)
0090 (2019-12-05): Nils-Christian Kempke (kempke@zib.de)
```



YOU NEVER WALK ALONE...

```
0091 (2020-01-23): Boro Sofranac (sofranac@zib.de)
0092 (2020-09-08): Mark (turner@zib.de)
0093 (2020-09-24): adm_timo (adm_timo@optmac01.zib.de)
0094 (2020-10-09): Jasper van Doornmalen (m.j.v.doornmalen@tue.nl)
0095 (2020-12-07): Erik Tadewaldt (tadewaldt@zib.de)
0096 (2021-01-07): Mathieu Besancon (mathieu.besancon@gmail.com)
0097 (2021-01-08): Philipp Wellner (wellner@zib.de)
0098 (2021-01-28): Alexander Hoen (hoen@zib.de)
0099 (2021-01-28): Turner (turner@zib.de)
0100 (2021-03-30): ahoen (hoen@zib.de)
0101 (2021-05-07): Didier Chetelat (didier.chetelat@polymtl.ca)
0102 (2021-10-01): Sander Borst (s.borst@live.nl)
0103 (2021-10-14): Gioni Mexi (mexi@zib.de)
0104 (2021-11-11): Jurgen Lentz (jurgen.lentz@rwth-aachen.de)
0105 (2021-11-12): Suresh Bolusani (sub214@lehigh.edu)
0106 (2022-04-12): Gioni (mexi@zib.de)
0107 (2022-07-04): Christoph Graczyk (graczyk@zib.de)
```



to all SCIP developers and users
for keeping the project alive!