Machine Learning for Automated Theorem Proving: Learning to Solve SAT and QSAT

Other titles in Foundations and Trends® in Machine Learning

Spectral Methods for Data Science: A Statistical Perspective
Yuxin Chen, Yuejie Chi, Jianqing Fan and Cong Ma
ISBN: 978-1-68083-896-1

Tensor Regression
Jiani Liu, Ce Zhu, Zhen Long and Yipeng Liu
ISBN: 978-1-68083-886-2

Minimum-Distortion Embedding
Akshay Agrawal, Alnur Ali and Stephen Boyd
ISBN: 978-1-68083-888-6

Graph Kernels: State-of-the-Art and Future Challenges
Karsten Borgwardt, Elisabetta Ghisu, Felipe Llinares-López, Leslie O'Bray and Bastian Rieck
ISBN: 978-1-68083-770-4

Data Analytics on Graphs Part III: Machine Learning on Graphs, from Graph Topology to Applications
Ljubiša Stanković, Danilo Mandic, Miloš Daković, Miloš Brajović, Bruno Scalzo, Shengxi Li and Anthony G. Constantinides
ISBN: 978-1-68083-982-16

Data Analytics on Graphs Part II: Signals on Graphso
Ljubiša Stanković, Danilo Mandic, Miloš Daković, Miloš Brajović, Bruno Scalzo, Shengxi Li and Anthony G. Constantinides
ISBN: 978-1-68083-982-1

Data Analytics on Graphs Part I: Graphs and Spectra on Graphs
Ljubiša Stanković, Danilo Mandic, Miloš Daković, Miloš Brajović, Bruno Scalzo, Shengxi Li and Anthony G. Constantinides
ISBN: 978-1-68083-982-1

Machine Learning for Automated Theorem Proving: Learning to Solve SAT and QSAT

Sean B. Holden
University of Cambridge
UK
sbh11@cl.cam.ac.uk

now
the essence of knowledge
Boston — Delft

Foundations and Trends® in Machine Learning

Published, sold and distributed by:
now Publishers Inc.
PO Box 1024
Hanover, MA 02339
United States
Tel. +1-781-985-4510
www.nowpublishers.com
sales@nowpublishers.com

Outside North America:
now Publishers Inc.
PO Box 179
2600 AD Delft
The Netherlands
Tel. +31-6-51115274

The preferred citation for this publication is

S.B. Holden. *Machine Learning for Automated Theorem Proving: Learning to Solve SAT and QSAT*. Foundations and Trends® in Machine Learning, vol. 14, no. 6, pp. 807–989, 2021.

ISBN: 978-1-68083-899-2

Foundations and Trends® in Machine Learning
Volume 14, Issue 6, 2021

Editorial Board

Editorial Scope

Topics

Foundations and Trends® in Machine Learning publishes survey and tutorial articles in the following topics:

- Adaptive control and signal processing
- Applications and case studies
- Behavioral, cognitive and neural learning
- Bayesian learning
- Classification and prediction
- Clustering
- Data mining
- Dimensionality reduction
- Evaluation
- Game theoretic learning
- Graphical models
- Independent component analysis
- Inductive logic programming
- Kernel methods
- Markov chain Monte Carlo
- Model choice
- Nonparametric methods
- Online learning
- Optimization
- Reinforcement learning
- Relational learning
- Robustness
- Spectral methods
- Statistical learning theory
- Variational inference
- Visualization

Information for Librarians

Foundations and Trends® in Machine Learning, 2021, Volume 14, 6 issues. ISSN paper version 1935-8237. ISSN online version 1935-8245. Also available as a combined paper and online subscription.

Contents

Machine Learning for Automated Theorem Proving: Learning to Solve SAT and QSAT

Sean B. Holden[1]

[1] *University of Cambridge, UK; sbh11@cl.cam.ac.uk*

ABSTRACT

The decision problem for Boolean satisfiability, generally referred to as SAT, is the archetypal NP-complete problem, and encodings of many problems of practical interest exist allowing them to be treated as SAT problems. Its generalization to quantified SAT (QSAT) is PSPACE-complete, and is useful for the same reason. Despite the computational complexity of SAT and QSAT, methods have been developed allowing large instances to be solved within reasonable resource constraints. These techniques have largely exploited algorithmic developments; however machine learning also exerts a significant influence in the development of state-of-the-art solvers. Here, the application of machine learning is delicate, as in many cases, even if a relevant learning problem can be solved, it may be that incorporating the result into a SAT or QSAT solver is counterproductive, because the run-time of such solvers can be sensitive to small implementation changes. The application of better machine learning methods in this area is thus an ongoing challenge, with characteristics unique to the field. This work provides a comprehensive review of the research to date on incorporating machine learning into SAT and QSAT solvers, as a resource for those interested in further advancing the field.

Sean B. Holden (2021), "Machine Learning for Automated Theorem Proving: Learning to Solve SAT and QSAT", Foundations and Trends® in Machine Learning: Vol. 14, No. 6, pp 807–989. DOI: 10.1561/2200000081.

1

Introduction

Automated theorem proving represents a significant and long-standing area of research in computer science, with numerous applications. A large proportion of the methods developed to date for the implementation of *automated theorem provers (ATPs)* have been algorithmic, sharing a great deal in common with the wider study of heuristic search algorithms (Harrison, 2009). However in recent years researchers have begun to incorporate *machine learning (ML)* methods (Murphy, 2012) into ATPs in an effort to extract better performance.

ATPs represent a compelling area in which to explore the application of ML. It is well-known that theorem-proving problems are computationally intractable, with the exception of specific, limited cases. Even in the apparently simple case of *propositional logic* the task is NP-hard, and adding *quantifiers* makes the task PSPACE-complete (Garey and Johnson, 1979). Taking a small step further we arrive at *first-order logic (FOL)*, which is undecidable (Boolos *et al.*, 2007). In addition to the general computational complexity of theorem-proving problems, they have a common property that makes them challenging as a target for ML: even the most trivial change to the statement of a problem can have a huge impact on the complexity of any subsequent proof attempt (Fuchs and Fuchs, 1998; Hutter *et al.*, 2007; Hutter *et al.*, 2009; Biere and Fröhlich, 2015; Biere and Fröhlich, 2019).

The aim of this work is to review the research that has appeared to date on incorporating ML methods into solvers for propositional *satisfiability (SAT)* problems, and also solvers for its immediate variants such as *quantified SAT (QSAT)*.

In a sense, these are some of the simplest possible ATP problems. (Any instance of a SAT problem can be represented as a Boolean formula in conjunctive normal form, and it is undeniably hard to propose anything much simpler.) But the combination of the computational challenges such problems present, and the enormous range of significant, practical applications that can be addressed this way, makes general solvers for SAT and its friends a compelling target for research. Marques-Silva (2008) reviews applications of SAT solvers circa 2008, and the interested reader might consult work applying them to bounded model checking (Biere *et al.*, 1999; Clarke *et al.*, 2001), planning (Kautz and Selman, 1992; Kautz, 2006), bioinformatics (Lynce and Marques-Silva, 2006; Graça *et al.*, 2010), allocation of radio spectrum (Fréchette *et al.*, 2016), and software verification (Babić and Hu, 2007). A further notable application has been the solution of the *Boolean Pythagorean triples* problem by Heule *et al.* (2016), resulting in what is currently considered the longest mathematical proof in history.

1.1 Coverage

Work on applying ML in this context appears to have started with Ertel *et al.* (1989) and Johnson (1989). At that time the limited availability of computing power and the limitations of existing solvers made the studies necessarily small by current standards, in terms of the size of the problems addressed, and also of the ML methods applied. This review is the result of a systematic search for literature appearing from then until late 2020.

SAT/QSAT solving and machine learning are both large and long-standing areas of research, and each has a correspondingly large literature. As these are two apparently rather unrelated fields, it is therefore inevitable that any reader versed in one might feel less confident with the other. (It has certainly been my experience in talking to researchers from both domains that this is often the case.) It would not be feasible to explain either, let alone both, areas in full detail here; and in any case, this is not intended to be a textbook on either subject. I have provided an introduction to each, but experts in either area might find one presentation overly elementary and the other too brief. The aim has been to provide sufficient information to make this work self-contained for both sides while maintaining a manageable length; however I expect that for many there will be areas where further reading will be necessary.

I wrote this work guided by two central aims for what the reader should gain from it. First, they should know *what has been tried*. In presenting the material, I concentrate on the learning methods used and the way in which they have been incorporated into solvers. As the literature rarely if ever presents methods not leading to performance improvements of some kind, less consideration is given to the details of the level of improvement achieved, because I believe such details are secondary to my second aim, which is: that the reader should understand the often *complex interaction between ATP and ML that is needed for success* in these undeniably challenging applications.

In order to achieve these aims it was necessary to be quite selective in the level of detail used to present various methods. Some research is presented in very great detail, relating to the learning method and its relationship with a solver, the description of the data used, or the experimental method employed. Other research is presented in less detail, although I hope at a level sufficient to allow the reader to understand what was done, and why. With the exception of the Chapters on ATP and ML, each Chapter presents a discussion summarizing what I believe are the central lessons to be taken from it. Where methods have been presented in greater detail, it is generally in the service of these arguments.

1.2 Outline of the review

Chapter 2 presents an introduction to the SAT problem, and to contemporary methods for its solution. Much of this section is devoted to summarizing the operation of *Conflict-Driven Clause Learning (CDCL)* solvers;[1] first, as these form the core of many of the most successful SAT solvers available; and second, because there are many distinct areas of their operation that have provided a point at which to introduce ML, and this therefore provides a road map for a large portion of the review. This section also briefly describes *portfolio solvers* and *local search* solvers, which have also been targets for ML, and which will be described further in later Chapters.

Chapter 3 provides a complementary introduction to some of the ML methods most commonly applied to SAT and QSAT solvers; this work spans supervised and unsupervised learning in addition to n-armed bandits, reinforcement learning, neural networks and evolutionary computing. In addition we describe some of the main sources of problems available for testing SAT and QSAT solvers; as these are often annotated such that we know which problems are satisfiable, and which are not, they provide a valuable resource for training ML methods.

Many applications of ML in this domain have required a phase of *feature engineering*, whereby a problem, typically expressed in *conjunctive normal form (CNF)*, is converted into a vector of real numbers suitable for use by an ML method. Chapter 4 reviews common sets of features that have been used, and that continue to form the basis for many ongoing studies. More recent work has made significant use of *graph neural networks* to (partially) automate the feature engineering process, and we introduce these here also.

There are, broadly-speaking, four ways in which ML has been applied to SAT solvers: by treating SAT directly as a classification problem; by building *portfolios* of existing SAT solvers; by modifying CDCL solvers; and by treating the problem as a form of *local search*.

In Chapter 5 we describe work aiming to identify satisfiability directly, without necessarily also obtaining a satisfying assignment of variables if one exists. Here, the SAT problem is treated as a classification

[1] There is an important distinction to be made here for the avoidance of confusion. The term ‘learning’ in the context of a CDCL solver is, at least at first glance, unconnected to the idea of machine learning. It is used to describe the addition of one or more new clauses to a problem after analysing a conflict during the search for a satisfying assignment; this is explained in more detail in Section 2.4.4. The use of the term ‘learning’ in both contexts is ubiquitous however, and we expand on the distinction a little further in Section 3.1.5.

problem: given a formula f, we aim to return the answer 'yes' or 'no', indicating whether or not the problem is satisfiable. In some cases it may be possible to extract a satisfying assignment as a side-effect.

Portfolio solvers are addressed in Chapter 6. Here, a collection of different SAT solvers is used in some combination to attack a problem. Chapter 7 then reviews the application of ML to CDCL solvers, addressing in turn the way in which ML has been applied to the individual elements described in Chapter 2. Chapter 8 describes the application of ML to local search SAT solvers.

In Chapter 9 we address attempts to introduce ML into solvers for QSAT. This area has received comparatively little attention, but work has appeared addressing ML for both portfolio solvers, and individual solvers.

While this review mainly addresses solvers for SAT and QSAT—these problems having received considerable attention as they have clear and significant applications—in Chapter 10 we briefly address machine learning applied to *intuitionistic propositional logic (IPL)* (Dalen, 2001). While this logic is of more foundational interest, having few applications beyond the philosophy of mathematics, it is related sufficiently closely to propositional logic that I feel attempts to apply machine learning to the search for proofs in IPL are relevant.

Chapter 11 concludes.

1.3 Limits to Coverage

A body of research exists addressing methods for automatically configuring algorithms that expose parameters—a process sometimes referred to as the *algorithm configuration problem*. Effective methods such as ParamILS (Hutter *et al.*, 2009) and, perhaps the best-known system of this kind, *Sequential Model-based Algorithm Configuration (SMAC)* (Hutter *et al.*, 2011), are now common. Algorithms in this class can clearly be applied to SAT/QSAT and related solvers, which invariably have parameters governing aspects of their operation. In compiling this review, I have aimed to focus on material that has a *specific emphasis* on SAT, QSAT and (closely) related problems. As a result, I decided not to describe in detail work such as that of Kadioglu *et al.* (2010) and Malitsky *et al.* (2013), that develops a general method for algorithm configuration and uses SAT as a test case, or Hutter *et al.* (2007) and Mangla *et al.* (2020), that is predominantly an application of an existing algorithm configuration method to SAT. For the same reasons, I have not included work that mainly relies on the application of general methods for selecting an algorithm from a collection of candidates; see Kotthoff (2016) for a review of such methods.

1.4 What Should the Reader Gain?

It is my hope that ML researchers might gain from this work, an understanding of state-of-the-art SAT and QSAT solvers that is sufficient to make new opportunities for applying their own ML research to this domain clearly visible. It is equally my hope that ATP researchers will gain a complementary understanding, giving them a clear appreciation of how state-of-the-art machine learning might help them to design better solvers. For both constituencies, I aim to show what has already been achieved at the time of writing, at a level of detail sufficient to provide a basis for new work.

Acknowledgements

In 2016 Josef Urban invited me to speak at the *1st Conference on Artificial Intelligence and Theorem Proving (AITP)*. I offered to give a survey talk on applications of machine learning to automated theorem provers. Having given the talk it seemed like a good idea to write it up in full.

I thought this would be a straightforward process, but it did not take long to discover that the full extent of the literature on the subject is genuinely impressive. In any case, here is the result.

Thanks for the invitation Josef.

I've done my share of reviewing, and I'm aware that reviewing a work of this length is a major undertaking. I therefore offer great thanks to the anonymous reviewer for their careful reading and numerous useful suggestions.

In reading the literature underlying this work there were inevitably occasions where I felt the need to contact the original authors for clarification. All responded quickly and helpfully. Thanks to all of them.

Appendices

A

Abbreviations

Abbreviation	Meaning
ATP	Automated theorem-prover
CAL	Clauses active list
CDCL	Conflict-Driven Clause Learning
CHB	Conflict history-based
CIG	Clause incidence graph
CNF	Conjunctive normal form
CNN	Convolutional neural network
CSP	Constraint satisfaction problem
CVIG	Clause-variable incidence graph
DAG	Directed acyclic graph
DPLL	Davis, Putnam, Logemann, Loveland
DRAT	Deletion Resolution Asymmetric Tautology
EHM	Empirical hardness model
EP	Evolutionary programming
ES	Evolutionary strategy

Continued overleaf...

Abbreviation	Meaning
ERWA	Exponential recency weighted average
EVSIDS	Exponential VSIDS
GA	Genetic algorithm
GLR	Global learning rate
GNN	Graph neural network
GP	Genetic program
IPL	Intuitionistic propositional logic
LBD	Literals blocks distance
LRB	Learning rate branching
LSTM	Long short-term memory
MAB	Multi-armed bandit
ML	Machine learning
MLB	Machine learning-based restart
MLP	Multi-layer perceptron
MPNN	Message-passing neural network
NN	Neural network
QSAT	Quantified satisfiability
RL	Reinforcement learning
SAT	Satisfiability
SVM	Support vector machine
SGDB	Stochastic Gradient Descent Branching
UC	Unsatisfiable core
UCB	Upper confidence bound
UIP	Unique implication point
VIG	Variable incidence graph
VSIDS	Variable State Independent Decaying Sum

B

Symbols

General	
$\mathbf{I}$	Identity matrix
$\mathbb{R}^i$	Set of i-dimensional vectors with real elements
v_i	Element i of a vector $\mathbf{v}$
$\mathbb{R}^{i \times j}$	Set of i by j matrices with real elements
$M_{i,j}$	Element at row i, column j of a matrix $\mathbf{M}$
$\mathbb{I}$	Indicator function: $\mathbb{I}[P]$ is 1 if P is true and 0 otherwise
$\mathbf{1}_{ij}$	i by j matrix with all elements equal to 1.
$N(\mathbf{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma})$	Multivariate normal density with mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Sigma}$
$\otimes$	Element-by-element multiplication of vectors
$[n]$	The set $\{1, \ldots, n\}$

Continued overleaf...

The SAT Problem	
V	Set of variables
C	Set of clauses
v	A variable. or |V|, according to context
c	A clause, or |C|, according to context
l	A literal
f, ϕ, ψ	Propositional formulas
A	Assignment
$a(v)$	Activity of a variable
$a(c)$	Activity of a clause

Machine Learning	
n	Dimension of feature space for a classifier
m	Size of training set
$\mathbf{s}$	Sequence containing m training examples
F	Function mapping instances of a problem to feature vectors
$\mathbb{A}$	Learning algorithm
$\mathcal{H}$	Hypothesis space
Z	Constant normalizing a probability distribution
C	Random variable denoting a class
$\mathbf{x}$	Instance vector
k	Dimension of the extended space
p	Number of basis functions
ϕ_i	Basis functions
λ	Regularization parameter
$\boldsymbol{\phi}(\mathbf{x})$	Mapping from instance $\mathbf{x}$ to the extended space
$\mathbf{\Phi}$	Matrix of $\boldsymbol{\phi}(\mathbf{x})$ for $\mathbf{x}$ in a training sequence
$\sigma(x)$	Step or sigmoid function
$\boldsymbol{\theta}$, $\mathbf{w}$	Vectors of parameters
K	Number of clusters

Continued overleaf...

Machine Learning	
r_i	Reward sequence
$r_{i,t}$	Reward from arm i of a multi-armed bandit at time t
α	EWMA discounting factor
γ	Bandit or reinforcement learning discount factor
$\hat{r}_T$	Estimated bandit reward at time T
$\mathcal{S}$	RL state set
$\mathcal{A}$	RL action set
p	RL policy
R	RL discounted reward
μ	Step size for gradient descent
c	Number of classes in a problem
$\mathbf{K}$	CNN kernel
t	Step in a sequence
T	Final step in a sequence
O	Objective function

References

Aksoy, L. and E. O. Gunes. (2005). "An Evolutionary Local Search Algorithm for the Satisfiability Problem". In: *Proceedings of the 14th Turkish Symposium on Artificial Intelligence and Neural Networks (TAINN)*. Ed. by F. A. Savakı. Vol. 3949. *Lecture Notes in Computer Science*. Springer. 185–193.

Allan, J. A. and S. Minton. (1996). "Selecting the Right Heuristic Algorithm: Runtime Performance Predictors". In: *Advances in Artificial Intelligence: 11th Biennial Conference of the Canadian Society for Computational Studies of Intelligence*. Ed. by G. McCalla. Vol. 1081. *Lecture Notes in Computer Science*. Springer. 41–53.

Amadini, R., M. Gabbrielli, and J. Mauro. (2015). "A Multicore Tool for Constraint Solving". In: *Proceedings of the 24th International Joint Conference on Artificial Intelligence (IJCAI)*. Ed. by Q. Yang and M. Wooldridge. AAAI Press/International Joint Conferences on Artificial Intelligence. 232–238.

Amizadeh, S., S. Matusevych, and M. Weimer. (2019). "Learning To Solve Circuit-SAT: An Unsupervised Differentiable Approach". In: *Proceedings of the International Conference on Learning Representations*.

Ansótegui, C., M. L. Bonet, J. Giráldez-Cru, and J. Levy. (2014). "The Fractal Dimension of SAT Formulas". In: *Proceedings of the 7th International Joint Conference on Automated Reasoning (IJCAR)*. Ed. by S. Demri, D. Kapur, and C. Weidenbach. Vol. 8562. *Lecture Notes in Computer Science*. Springer. 107–121.

Ansótegui, C., M. L. Bonet, J. Giráldez-Cru, and J. Levy. (2017). "Structure instances for SAT instances classification". *Journal of Applied Logic*. 23(Sept.): 27–39.

Ansótegui, C., J. Giráldez-Cru, and J. Levy. (2012). "The Community Structure of SAT Formulas". In: *Proceedings of the 15th International Conference on Theory and Applications of Satisfiability Testing*. Ed. by A. Cimatti and R. Sebastiani. Vol. 7317. *Lecture Notes in Computer Science*. Springer. 410–423.

Ansótegui, C., M. Sellmann, and K. Tierney. (2009). "A Gender-Based Genetic Algorithm for the Automatic Configuration of Algorithms". In: *Proceedings of the 15th International Conference on Principles and Practice of Constraint Programming (CP)*. Ed. by I. P. Gent. Vol. 5732. *Lecture Notes in Computer Science*. Springer. 142–157.

Anthony, M. and P. L. Bartlett. (2009). *Pattern Recognition and Machine Learning*. Cambridge University Press.

Audemard, G. and L. Simon. (2018). "On the Glucose SAT Solver". *International Journal on Artificial Intelligence Tools*. 27(1).

Audemard, G. and L. Simon. (2009). "Predicting learnt clauses quality in modern SAT solvers". In: *Proceedings of the 21st International Joint Conference on Artificial Intelligence (IJCAI)*. Morgan Kaufmann. 399–404.

Auer, P., N. Cesa-Bianci, Y. Freund, and R. E. Schapire. (1995). "Gambling in a rigged casino: The adversarial multi-armed bandit problem". In: *Proceedings of the 36th IEEE Annual Symposium on Foundations of Computer Science*. IEEE. 332–331.

Ba, J. L., J. R. Kiros, and G. E. Hinton. (2016). "Layer Normalization". arXiv: 1607.06450v1.

Babić, D. and A. J. Hu. (2007). "Structural Abstraction of Software Verification Conditions". In: *Proceedings of the 19th International Conference on Computer Aided Verification (CAV)*. Ed. by W. Damm and H. Hermanns. Vol. 4590. *Lecture Notes in Computer Science*. Springer. 366–378.

Bader-El-Den, M. and R. Poli. (2007). "Generating SAT Local Search Heuristics Using a GP Hyper-Heuristic Framework". In: *Proceedings of the 8th International Conference on Artificial Evolution*. Ed. by N. Monmarché, E.-G. Talbi, P. Collet, M. Schoenauer, and E. Lutton. Vol. 4926. *Lecture Notes in Computer Science*. Springer. 37–49.

Bader-El-Den, M. and R. Poli. (2008a). "Analysis and extension of the Inc* on the satisfiability testing problem". In: *Proceedings of the IEEE Congress on Evolutionary Computation*. IEEE. 3342–3349.

Bader-El-Den, M. and R. Poli. (2008b). "Evolving Effective Incremental Solvers for SAT with a Hyper-Heuristic Framework Based on Genetic Programming". In: *Genetic Programming Theory and Practice VI*. Ed. by B. Worzel, T. Soule, and R. Riolo. *Genetic and Evolutionary Computation*. Springer. 1–16.

Bader-El-Den, M. and R. Poli. (2008c). "Inc*: An Incremental Approach for Improving Local Search Heuristics". In: *Proceedings of the 8th European Conference on Evolutionary Computation in Combinatorial Optimization (EvoCOP)*. Ed. by J. van Hemert and C. Cotta. Vol. 4972. *Lecture Notes in Computer Science*. Springer. 194–205.

Bain, S., J. Thornton, and A. Sattar. (2005a). "A Comparison of Evolutionary Methods for the Discovery of Local Search Heuristics". In: *Proceedings of the 18th Australasian Joint Conference on Artificial Intelligence*. Ed. by S. Zhang and R. Jarvis. Vol. 3809. *Lecture Notes in Computer Science*. Springer. 1068–1074.

Bain, S., J. Thornton, and A. Sattar. (2005b). "Evolving Variable-Ordering Heuristics for Constrained Optimisation". In: *Proceedings of the 11th International Conference on Principles and Practice of Constraint Programming*. Ed. by P. van Beek. Vol. 3709. *Lecture Notes in Computer Science*. Springer. 732–736.

Bertels, A. R. and D. R. Tauritz. (2016). "Why Asynchronous Parallel Evolution is the Future of Hyper-heuristics: A CDCL SAT Solver Case Study". In: *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO)*. Ed. by T. Friedrich. Association for Computing Machinary. 1359–1365.

Bertels, A. R. (2016). "Automated design of boolean satisfiability solvers employing evolutionary computation". *MA thesis*. Missouri Institute of Science and Technology. 7549.

Biere, A. (2008a). "Adaptive Restart Strategies for Conflict Driven SAT Solvers". In: *Proceedings of the 11th International Conference on Theory and Applications of Satisfiability Testing*. Ed. by H. K. Büning and X. Zhao. Vol. 4996. *Lecture Notes in Computer Science*. Springer. 28–33.

Biere, A. (2008b). "PicoSAT Essentials". *Journal of Satisfiability, Boolean Modeling and Computation*. 4(2-4): 75–97.

Biere, A., A. Cimatti, E. Clarke, and Y. Zhu. (1999). "Symbolic Model Checking without BDDs". In: *Proceedings of the 5th International Conference on Construction and Analysis of Systems (TACAS)*. Ed. by W. R. Cleaveland. Vol. 1579. *Lecture Notes in Computer Science*. Springer. 197–207.

Biere, A., K. Fazekas, M. Fleury, and M. Heisinger. (2020). "CaDiCaL, Kissat, Paracooba, Plingeling and Treengeling Entering the SAT Competition 2020". In: *Proceedings of SAT Competition 2020: Solver and Benchmark Descriptions*. Ed. by T. Balyo, N. Froleyks, M. J. Heule, M. Iser, M. Järvisalo, and M. Suda. *Department of Computer Science Report Series B*. No. B-2020-1. Department of Computer Science, University of Helsinki.

Biere, A. and A. Fröhlich. (2015). "Evaluating CDCL Variable Scoring Schemes". In: *Proceedings of the 18th International Conference on Theory and Applications of Satisfiability Testing*. Ed. by M. Heule and S. Weaver. Vol. 9340. *Lecture Notes in Computer Science*. Springer. 405–422.

Biere, A. and A. Fröhlich. (2019). "Evaluating CDCL Restart Schemes". In: *Proceedings of Pragmatics of SAT 2015 and 2018*. Vol. 59. *EPiC Series in Computing*. EasyChair. 1–17.

Biere, A., M. Huele, H. van Maaren, and T. Walsh. (2009). *Handbook of Satisfiability*. Vol. 85. *Frontiers in Artificial Intelligence and Applications*. IOS Press.

Bischl, B., M. Binder, M. Lang, T. Pielok, J. Richter, S. Coors, J. Thomas, T. Ullmann, M. Becker, A.-L. Boulesteix, D. Deng, and M. Lindauer. (2021). "Hyperparameter Optimization: Foundations, Algorithms, Best Practices and Open Challenges". arXiv: 2107.05847v2 [stat.ML].

Bishop, C. M. (2006). *Pattern Recognition and Machine Learning*. Springer.

Blanchette, J. C., M. Fleury, P. Lammich, and C. Weidenbach. (2018). "A Verified SAT Solver Framework with Learn, Forget, Restart and Incrementality". *Journal of Automated Reasoning*. 61(1–5): 333–365.

Blondel, V. D., J.-L. Guillaume, R. Lambiotte, and E. Lefebvre. (2008). "Fast unfolding of communities in large networks". *Journal of Statistical Mechanics: Theory and Experiment*. 2008(10).

Boolos, G. S., J. P. Burgess, and R. C. Jeffrey. (2007). *Computability and Logic*. 5th Edition. Cambridge University Press.

Boyan, J. A. (1998). "Learning Evaluation Functions for Global Optimization". *PhD thesis*. Pittsburgh, PA 15213: Carnegie Mellon University. CMU-CS-98-152.

Boyan, J. A. and A. W. Moore. (1998). "Learning Evaluations Functions for Global Optimization and Boolean Satisfiability". In: *Proceedings of the 15th National Conference on Artificial Intelligence (AAAI)*. 3–10.

Boyan, J. A. and A. W. Moore. (2000). "Learning Evaluation Functions to Improve Optimization by Local Search". *Journal of Machine Learning Research*. 1: 77–112.

Breiman, L. (2001). "Random Forests". *Machine Learning*. 45(1): 5–32.

Bridge, J. P., S. B. Holden, and L. C. Paulson. (2014). "Machine Learning for First-Order Theorem Proving: Learning to Select a Good Heuristic". *Journal of Automated Reasoning*. 53(Feb.): 141–172.

Bünz, B. and M. Lamm. (2017). "Graph Neural Networks and Boolean Satisfiability". arXiv: 1702.03592 [cs.AI].

Cameron, C., R. Chen, J. Hartford, and K. Leyton-Brown. (2020). "Predicting Propositional Satisfiabilty via End-to-End Learning". In: *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI-20)*. Vol. 34. No. 4. AAAI Press.

Cameron, C., H. H. Hoos, K. Leyton-Brown, and F. Hutter. (2017). "OASC-2017: *Zilla Submission". In: *Proceedings of Machine Learning Research*. Ed. by M. Lindauer, J. N. van Rijn, and L. Kotthoff. Vol. 79. 15–18.

Carvalho, E. and J. Marques-Silva. (2004). "Using Rewarding Mechanisms for Improving Branching Heuristics". In: *Proceedings of the 7th International Conference on Theory and Applications of Satisfiability Testing*.

Chang, W., G. Wu, and Y. Xu. (2017). "Adding a LBD-based Rewarding Mechanism in Branching Heuristic for SAT Solvers". In: *Proceedings of the 12th International Conference on Intelligent Systems and Knowledge Engineering (ISKE)*.

Chang, W., Y. Xu, and S. Chen. (2018). "A New Rewarding Mechanism for Branching Heuristic in SAT Solvers". *International Journal of Computational Intelligence Systems*. 12(1): 334–341.

Chen, W., A. Howe, and D. Whitley. (2014). "MiniSAT with Classification-based Preprocessing". In: *Proceedings of SAT Competition 2014: Solver and Benchmark Descriptions*. Ed. by A. Belov, D. Diepold, M. J. Heule, and M. Järvisalo. *Department of Computer Science Report Series B*. No. B-2014-2. Department of Computer Science, University of Helsinki. 41–42.

Chen, Z. and Z. Yang. (2019). "Graph Neural Reasoning May Fail in Certifying Boolean Unsatisfiability". arXiv: 1909.11588 [cs.LG].

Chu, G., A. Harwood, and P. J. Stuckey. (2010). "Cache Conscious Data Structures for Boolean Satisfiability Solvers". *Journal of Satisfiability, Boolean Modeling and Computation*. 6(1-3): 99–120.

Chvalovský, K. (2019). "Top-Down Neural Model For Formulae". In: *Proceedings of the International Conference on Learning Representations*.

Clarke, E., A. Biere, R. Raimi, and Y. Zhu. (2001). "Bounded Model Checking Using Satisfiability Solving". *Formal Methods in System Design*. 19: 7–34.

Clauset, A., C. R. Shalizi, and M. E. J. Newman. (2009). “Power-Law Distributions in Empirical Data”. *SIAM Review*. 51(4): 661–703.

Dalen, D. van. (2001). “Intuitionistic Logic”. In: *The Blackwell Guide to Philosophical Logic*. Ed. by L. Goble. Blackwell Publishers. Chap. 11. 224–257.

Daumé, H. and D. Marcu. (2005). “Learning as Search Optimization: Approximate Large Margin Methods for Structured Prediction”. In: *Proceedings of the 22nd International Conference on Machine Learning (ICML)*. 169–176.

Davies, M., G. Logemann, and D. Loveland. (1962). “A machine program for theorem-proving”. *Communications of the ACM*. 5(7): 394–397.

Dershowitz, N., Z. Hanna, and J. Katz. (2005). “Bounded Model Checking with QBF”. In: *Proceedings of the International Conference on Theory and Applications of Satisfiability Testing*. Ed. by F. Bacchus and T. Walsh. Vol. 3569. *Lecture Notes in Computer Science*. 408–414.

Devlin, D. and B. O’Sullivan. (2008). “Satisfiability as a Classification Problem”. In: *Proceedings of the 19th Irish Conference on Artificial Intelligence and Cognitive Science*. Ed. by D. Bridge, K. Brown, B. O’Sullivan, and H. Sorensen. University College Cork.

Devroye, L., L. Györfi, and G. Lugosi. (1996). *A Probabilistic Theory of Pattern Recognition*. Vol. 31. *Stochastic Modelling and Applied Probability*. Springer.

Duda, R. O., P. E. Hart, and D. G. Stork. (2000). *Pattern Classification*. 2nd Edition. Wiley.

Dyckhoff, R. (1992). “Contraction-Free Sequent Calculi for Intuitionistic Logic”. *The Journal of Symbolic Logic*. 57(3): 795–807.

Eén, N. and A. Biere. (2005). “Effective Preprocessing in SAT Through Variable and Clause Elimination”. In: *Proceedings of the 8th International Conference on Theory and Applications of Satisfiability Testing*. Ed. by F. Bacchus and T. Walsh. Vol. 3569. *Lecture Notes in Computer Science*. Springer. 61–75.

Eén, N. and N. Sörensson. (2003). “An Extensible SAT-solver”. In: *Proceedings of the 6th International Conference on Theory and Applications of Satisfiability Testing*. Ed. by E. Giunchiglia and A. Tacchella. Vol. 2919. *Lecture Notes in Computer Science*. Springer. 502–518.

Egly, U., T. Eiter, H. Tompits, and S. Woltran. (2000). “Solving Advanced Reasoning Tasks using Quantified Boolean Formulas”. In: *Proceedings of the Seventeenth National Conference on Artificial Intelligence*. The AAAI Press. 417–422.

Emmerich, M., O. M. Shir, and H. Wang. (2018). “Evolution Strategies”. In: *Handbook of Heuristics*. Ed. by R. Marti, P. Panos, and M. G. C. Resende. Springer. 1–31.

Engel, A. and C. V. den Broeck. (2001). *Statistical Mechanics of Learning*. Cambridge University Press.

Ertel, W., J. M. P. Schumann, and C. B. Suttner. (1989). “Learning Heuristics for a Theorem Prover using Back Propagation”. In: *5. Osterreichische Aertificial-Intelligence-Tagung*. Ed. by J. Retti and K. Leidlmair. Vol. 208. *Informatik-Fachbericht*. 87–95.

Evans, R., D. Saxton, D. Amos, P. Kohli, and E. Grefenstette. (2018). “Can Neural Networks Understand Logical Entailment?” In: *Proceedings of the 6th International Conference on Learning Representations*.

Färber, M., C. Kaliszyk, and J. Urban. (2021). “Machine Learning Guidance for Connection Tableaux”. *Journal of Automated Reasoning*. 65: 287–320.

Fernández-Delgado, M., E. Cernadas, S. Barro, and D. Amorim. (2014). “Do we Need Hundreds of Classifiers to Solve Real World Classification Problems?” *Journal of Machine Learning Research*. 15: 3133–3181.

Ferrari, M., C. Fiorentini, and G. Fiorino. (2010). “fCube: An Efficient Prover for Intuitionistic Propositional Logic”. In: *Proceedings of the 17th International Conference on Logic for Programming Artificial Intelligence and Reasoning (LPAR)*. Ed. by C. G. Fermüller and A. Voronkov. Vol. 6397. *Lecture Notes in Computer Science*. Springer. 294–301.

Fink, M. (2007). "Online Learning of Search Heuristics". *Proceedings of Machine Learning Research*. 2: 115–122.

Fleury, M., J. C. Blanchette, and P. Lammich. (2018). "A verified SAT solver with watched literals using imperative HOL". In: *Proceedings of the 7th ACM SIGPLAN International Conference on Certified Programs and Proof*. 158–171.

Flint, A. and M. B. Blaschko. (2012). "Perceptron Learning of SAT". In: *Proceedings of the 25th International Conference on Neural Information Processing (NIPS)*. Ed. by F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger. Vol. 2. Curran Associates Inc. 2771–2779.

Fréchette, A., N. Newman, and K. Leyton-Brown. (2016). "Solving the Station Repacking Problem". In: *Proceedings of the 30th AAAI Conference on Artificial Intelligence*. 702–709.

Fuchs, M. and M. Fuchs. (1998). "Feature-based learning of search-guiding heuristics for theorem proving". *AI Communications*. 11(3,4): 175–189.

Fukunaga, A. S. (2002). "Automated Discovery of Composite SAT Variable-Selection Heuristics". In: *Proceedings of the 18th National Conference on Artificial Intelligence (AAAI)*. The AAAI Press. 641–648.

Fukunaga, A. S. (2004). "Evolving Local Search Heuristics for SAT Using Genetic Programming". In: *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO)*. Ed. by K. Deb. Vol. 3103. *Lecture Notes in Computer Science*. Springer. 483–494.

Fukunaga, A. S. (2008). "Automated Discovery of Local Search Heuristics for Satisfiability Testing". *Evolutionary Computation*. 16(1): 31–61.

Fukunaga, A. S. (2009). "Massively Parallel Evolution of SAT Heuristics". In: *Proceedings of the IEEE Congress on Evolutionary Computation*. 1478–1485.

Gagliolo, M. and J. Schmidhuber. (2007). "Learning Restart Strategies". In: *Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI)*. 792–797.

Garey, M. R. and D. S. Johnson. (1979). *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman and Company.

Garivier, A. and E. Moulines. (2011). "On Upper-Confidence Bound Policies for Switching Bandit Problems". In: *Proceedings of the 22nd International Conference on Algorithmic Learning Theory*. Ed. by J. Kivinen, C. Szepesvári, E. Ukkonen, and T. Zeugmann. Vol. 6925. *Lecture Notes in Computer Science*. Springer. 174–188.

Gilmer, J., S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl. (2017). "Neural message passing for Quantum chemistry". In: *Proceedings of the 34th International Conference on Machine Learning (ICML)*. Vol. 70. 1263–1272.

Giunchiglia, E., M. Narizzano, L. Pulina, and A. Tacchella. (2005). "Quantified Boolean Formulas satisfiability library (QBFLIB)". URL: www.qbflib.org.

Giunchiglia, E., M. Narizzano, and A. Tacchella. (2006). "Clause/Term Resolution and Learning in the Evaluation of Quantified Boolean Formulas". *Journal of Artificial Intelligence Research*. 26(Aug.): 371–416.

Goldberg, E. and Y. Novikov. (2002). "BerkMin: A fast and robust SAT solver". In: *Proceedings of the 2002 Design, Automation and Test in Europe Conference and Exhibition*. IEEE. 142–149.

Gomes, C. P. and B. Selman. (2001). "Algorithm portfolios". *Artificial Intelligence*. 126: 43–62.

Goodfellow, I., Y. Bengio, and A. Courville. (2016). *Deep Learning*. MIT Press.

Gottlieb, J., E. Marchiori, and C. Rossi. (2002). "Evolutionary Algorithms for the Satisfiability Problem". *Evolutionary Computation*. 10(1): 35–50.

Graça, A., J. Marques-Silva, and I. Lynce. (2010). "Haplotype Inference Using Propositional Satisfiability". In: *Mathematical Approaches to Polymer Sequence Analysis and Related Problems*. Ed. by R. Bruni. Springer. 127–147.

Grozea, C. and M. Popescu. (2014). "Can Machine Learning Learn a Decision Oracle for NP Problems? A Test on SAT". *Fundamenta Informaticae*. 131: 441–450.

Guyon, I., S. Gunn, M. Nikravesh, and L. A. Zadeh, eds. (2006). *Feature Extraction: Foundations and Applications. Studies in Fuzziness and Soft Computing*. Springer.

Haim, S. and T. Walsh. (2008). "Online Estimation of SAT Solving Runtime". In: *Proceedings of the 11th International Conference on Theory and Applications of Satisfiability Testing*. Ed. by H. K. Büning and X. Zhao. Vol. 4996. *Lecture Notes in Computer Science*. Springer. 133–138.

Haim, S. and T. Walsh. (2009). "Restart Strategy Selection Using Machine Learning Techniques". In: *Proceedings of the 12th International Conference on Theory and Applications of Satisfiability Testing*. Ed. by O. Kullmann. Vol. 5584. *Lecture Notes in Computer Science*. Springer. 312–325.

Hamadi, Y., S. Jabbour, and L. Saïs. (2010). "Learning for Dynamic Subsumption". *International Journal on Artificial Intelligence Tools: Architectures, Languages, Algorithms*. 19(4): 511–529.

Hamilton, W. L. (2020). "Graph Representation Learning". *Synthesis Lectures on Artificial Intelligence and Machine Learning*. 14(3): 1–159.

Han, H. and F. Somenzi. (2009). "On-The-Fly Clause Improvement". In: *Proceedings of the 12th International Conference on Theory and Applications of Satisfiability Testing*. Ed. by O. Kullmann. Vol. 5584. *Lecture Notes in Computer Science*. Springer. 209–222.

Han, J. M. (2020a). "Enhancing SAT solvers with glue variable predictions". arXiv: 2007.02559v1 [cs.LO].

Han, J. M. (2020b). "Learning cubing heuristics for SAT from DRAT proofs". In: *Conference on Artificial Intelligence and Theorem Proving (AITP)*.

Harrison, J. (2009). *Handbook of Practical Logic and Automated Reasoning*. Cambridge University Press.

Hartford, J., D. Graham, K. Leyton-Brown, and S. Ravanbakhsh. (2018). "Deep Models of Interactions Across Sets". In: *Proceedings of the 35th International Conference on Machine Learning*. Vol. 80. *Proceedings of Machine Learning Research*. 1909–1918.

Hastie, T., R. Tibshirani, and J. Friedman. (2009). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. 2nd Edition. Springer.

Heule, M., M. Järvisalo, and M. Suda. (2019). *The international SAT Competitions web page*. URL: http://www.satcompetition.org/.

Heule, M. J. H., O. Kullmann, and V. W. Marek. (2016). "Solving and Verifying the Boolean Pythagorean Triples Problem via Cube-And-Conquer". In: *Proceedings of the 19th International Conference on Theory and Applications of Satisfiability Testing*. Ed. by N. Creignou and D. L. Berre. Vol. 9710. *Lecture Notes in Computer Science*. Springer. 228–245.

Heule, M. J. H., O. Kullmann, S. Wieringa, and A. Biere. (2011). "Cube and Conquer: Guiding CDCL SAT Solvers by Lookaheads". In: *Proceedings of the 7th International Haifa Verification Conference*. Ed. by K. Eder, J. Lourenço, and O. Shehory. Vol. 7261. *Lecture Notes in Computer Science*. Springer. 50–65.

Heule, M. J., O. Kullmann, and V. W. Marek. (2017). "Solving Very Hard Problems: Cube-and-Conquer, a Hybrid SAT Solving Method". In: *Proceedings of the 26th International Conference on Artificial Intelligence (IJCAI)*. Ed. by C. Sierra. 4864–4868.

Hochreiter, S. and J. Schmidhuber. (1997). "Long Short-Term Memory". *Neural Computation*. 9(8): 1735–1780.

Holldobler, S., N. Manthey, V. H. Nguyen, J. Stecklina, and P. Steinke. (2011). "A short overview of modern parallel SAT-solvers". In: *Proceedings of the International Conference on Computer Science and Information Systems*. IEEE. 201–206.

Holte, R. C. (1993). "Very Simple Classification Rules Perform Well on Most Commonly Used Datasets". *Machine Learning*. 11: 63–91.

Hoos, H., T. Peitl, F. Slivovsky, and S. Szeider. (2018). "Portfolio-Based Algorithm Selection for Circuit QBFs". In: *Proceedings of the 24th International Conference on Principles and Practice of Constraint Programming (CP)*. Ed. by J. Hooker. Vol. 11008. *Lecture Notes in Computer Science*. Springer. 195–209.

Hoos, H. H. (1999). "On the Run-Time Behavious of Stochastic Local Search Algorithms for SAT". In: *Proceedings of the 16th National Conference on Artificial Intelligence*. Association for the Advancement of Artificial Intelligence. AAAI Press. 661–666.

Hoos, H. H. and T. Stützle. (2000). "SATLIB: An Online Resource for Research on SAT". In: *SAT2000: Highlights of Satisfiability Research in the Year 2000*. Ed. by I. P. Gent, H. V. Maaren, and T. Walsh. Vol. 63. *Frontiers in Artificial Intelligence and Applications*. IOS Press. 283–292.

Hoos, H. H. and T. Stützle. (2019). *SATLIB—The Satisfiability Library*. URL: https://www.cs.ubc.ca/~hoos/SATLIB/.

Hopfield, J. J. and D. W. Tank. (1985). "'Neural' Computation of Decisions in Optimization Problems". *Biological Cybernetics*. 52(July): 141–152.

Hu, Y., X. Si, C. Hu, and J. Zhang. (2019). "A Review of Recurrent Neural Networks: LSTM Cells and Network Architectures". *Neural Computation*. 31: 1235–1270.

Huberman, B. A., R. M. Lukose, and T. Hogg. (1997). "An Economics Approach to Hard Computational Problems". *Science*. 275(Jan.): 51–54.

Hutter, F., D. Babić, H. H. Hoos, and A. J. Hu. (2007). "Boosting Verification by Automatic Tuning of Decision Procedures". In: *Proceedings of the 7th International Conference on Formal Methods in Computer-Aided Design*. IEEE. 27–34.

Hutter, F., Y. Hamadi, H. H. Hoos, and K. Leyton-Brown. (2006). "Performance Prediction and Automated Tuning of Randomized and Parametric Algorithms". In: *Proceedings of the 12th International Conference on Principles and Practice of Constraint Programming*. Ed. by F. Benhamou. Vol. 4204. *Lecture Notes in Computer Science*. Springer. 213–228.

Hutter, F., H. H. Hoos, and K. Leyton-Brown. (2011). "Sequential Model-Based Optimization for General Algorithm Configuration". In: *Proceedings of the 5th International Conference on Learning and Intelligent Optimization (LION)*. Ed. by C. A. C. Coello. Vol. 6683. *Lecture Notes in Computer Science*. Springer. 507–523.

Hutter, F., H. H. Hoos, K. Leyton-Brown, and T. Stützle. (2009). "ParamILS: An Automatic Algorithm Configuration Framework". *Journal of Artificial Intelligence Research*. 36(Oct.): 267–306.

Hutter, F., L. Kotthoff, and J. Vanschoren, eds. (2019). *Automated Machine Learning: Methods, Systems, Challenges*. *The Springer Series on Challenges in Machine Learning*. Springer.

Hutter, F., M. Lindauer, A. Balint, S. Bayless, H. Hoos, and K. Leyton-Brown. (2017). "The Configurable SAT Solver Challenge (CSSC)". *Artificial Intelligence*. 243: 1–25.

Hutter, F., D. A. D. Tompkins, and H. H. Hoos. (2002). "Scaling and Probabilistic Smoothing: Efficient Dynamic Local Search for SAT". In: *Proceedings of the 8th International Conference on Principles and Practice of Constraint Programming*. Ed. by P. V. Hentenryck. Vol. 2470. *Lecture Notes in Computer Science*. Springer. 233–248.

Illetskova, M., A. R. Bertels, J. M. Tuggle, A. Harter, S. Richter, D. R. Tauritz, S. Mulder, D. Bueno, M. Leger, and W. M. Siever. (2017). "Improving performance of CDCL SAT solvers by automated design of variable selection heuristics". In: *Proceedings of the IEEE Symposium Series on Computational Intelligence (SSCI)*. IEEE. 617–624.

Janota, M. (2018). "Towards Generalization in QBF Solving via Machine Learning". In: *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence (AAAI-18)*. AAAI Press. 6607–6614.

Janota, M., W. Klieber, J. Marques-Silva, and E. Clarke. (2016). "Solving QBF with counterexample guided refinement". *Artificial Intelligence*. 234: 1–25.

Järvisalo, M., M. J. H. Heule, and A. Biere. (2012). "Inprocessing Rules". In: *Proceedings of the 6th International Joint Conference on Automated Reasoning (IJCAR)*. Ed. by B. Gramlich, D. Miller, and U. Sattler. Vol. 7364. *Lecture Notes in Computer Science*. Springer. 355–370.

Jaszczur, S., M. Łuszczyk, and H. Michalewski. (2019). "Neural heuristics for SAT solving". In: *Proceedings of the 7th International Conference on Learning Representations*.

Jeroslow, R. G. and J. Wang. (1990). “Solving propositional satisfiability problems”. *Annals of Mathematics and Artificial Intelligence*. 1(1–4): 167–187.

Johnson, J. L. (1989). “A Neural Network Approach to the 3-Satisfiability Problem”. *Journal of Parallel and Distributed Computing*. 6: 435–449.

Jordan, C. and Ł. Kaiser. (2013). “Experiments with Reduction Finding”. In: *Proceedings of the 16th International Conference on Theory and Applications of Satisfiability Testing (SAT)*. Ed. by M. Järvisalo and A. V. Gelder. Vol. 7962. *Lecture Notes in Computer Science*. Springer. 192–207.

Kadioglu, S., Y. Malitski, A. Sabharwal, H. Samulowitz, and M. Sellmann. (2011). “Algorithm Selection and Scheduling”. In: *Proceedings of the 17th International Conference on Principles and Practice of Constraint Programming*. Ed. by J. Lee. Vol. 6876. *Lecture Notes in Computer Science*. Springer. 454–469.

Kadioglu, S., Y. Malitsky, M. Sellman, and K. Tierney. (2010). “ISAC—Instance-Specific Algorithm Configuration”. In: *Proceedings of the 19th European Conference on Artificial Intelligence (ECAI)*. 751–756.

Kaplan, E. L. and P. Meier. (1958). “Nonparametric Estimation from Incomplete Observations”. *Journal of the American Statistical Association*. 53(282): 457–481.

Kaufman, L. and P. J. Rousseeuw. (1990). *Finding Groups in Data: An Introduction to Cluster Analysis. Wiley Series in Probability and Statistics*. John Wiley & Sons, Inc.

Kautz, H. and B. Selman. (1992). “Planning as Satisfiability”. In: *Proceedings of the 10th European Conference on Articifial Intelligence (ECAI)*. Wiley. 359–363.

Kautz, H. A. (2006). “Deconstructing Planning as Satisfiability”. In: *Proceedings of the 21st National Conference on Artificial Intelligence (AAAI)*. Vol. 2. 1524–1526.

Khudabukhsh, A. R., L. Xu, H. H. Hoos, and K. Leyton-Brown. (2016). “SATenstein: Automatically building local search SAT solvers from components”. *Artificial Intelligence*. 232: 20–42.

Kibria, R. H. (2007). "Evolving a Neural Network-Based Decision and Search Heuristic for DPLL SAT Solvers". In: *Proceedings of the International Joint Conference on Neural Networks (IJCNN)*. 765–770.

Kibria, R. H. and Y. Li. (2006). "Optimizing the Initialization of Dynamic Decision Heuristics in DPLL SAT Solvers Using Genetic Programming". In: *Proceedings of the 9th European Conference on Genetic Programming (EuroGP)*. Ed. by P. Collett, M. Tomassini, M. Ebner, S. Gustafson, and A. Ekárt. Vol. 3905. *Lecture Notes in Computer Science*. Springer. 331–340.

Kibria, R. H. (2011). "Soft Computing Approaches to DPLL SAT Solver Optimization". *PhD thesis*. Technische Universität Darmstadt.

Kingma, D. P. and J. Ba. (2015). "Adam: A Method For Stochastic Optimization". In: *Proceedings of the International Conference on Learning Representations*.

Klieber, W., S. Sapra, S. Gao, and E. Clarke. (2010). "A Non-prenex, Non-clausal QBF Solver with Game-State Learning". In: *Proceedings of the 13th International Conference on Theory and Applications of Satisfiability Testing*. Ed. by O. Strichman and S. Szeider. Vol. 6175. *Lecture Notes in Computer Science*. Springer. 128–142.

Kohavi, R. (1995). "A study of cross-validation and bootstrap for accuracy estimation and model selection". In: *Proceedings of the 14th International Joint Conference on Artificial Intelligence (IJCAI)*. Vol. 2. Morgan Kaufmann. 1137–1143.

Kotthoff, L. (2016). "Algorithm Selection for Combinatorial Search Problems: A Survey". In: *Data Mining and Constraint Programming: Foundations of a Cross-Disciplinary Approach*. Ed. by C. Bessiere, L. D. Raedt, L. Kotthoff, S. Nijssen, B. O'Sullivan, and D. Pedreschi. Vol. 10101. *Lecture Notes in Computer Science*. Springer. 149–190.

Koza, J. R. (1992). *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. The MIT Press.

Kurin, V., S. Godil, S. Whiteson, and B. Catanzaro. (2019). "Improving SAT Solver Heuristics with Graph Networks and Reinforcement Learning". arXiv: 1909.11830 [cs.LG].

Kusumoto, M., K. Yahata, and M. Sakai. (2018). "Automated Theorem Proving in Intuitionistic Propositional Logic by Deep Reinforcement Learning". arXiv: 1811.00796 [cs.LG].

Laarhoven, P. J. van and E. H. Aarts. (1987). *Simulated Annealing: Theory and Applications*. Springer.

Lagoudakis, M. G. and M. L. Littman. (2000). "Algorithm Selection using Reinforcement Learning". In: *Proceedings of the 17th International Conference on Machine Learning (ICML)*. Morgan Kaufmann. 511–518.

Lagoudakis, M. G. and M. L. Littman. (2001). "Learning to Select Branching Rules in the DPLL Procedure for Satisfiability". *Electronic Notes in Discrete Mathematics*. 9(June): 344–359.

Lederman, G., M. N. Rabe, E. A. Lee, and S. A. Seshia. (2019). "Learning Heuristics for Quantified Formulas through Deep Reinforcement Learning". arXiv: 1807.08058v3 [cs.LO].

Letz, R., J. Schumann, S. Bayeri, and W. Bibel. (1992). "SETHEO: A high-performance theorem prover". *Journal of Automated Reasoning*. 8(2): 183–212.

Li, C.-M., F. Xiao, M. Luo, F. Manyà, Z. Lü, and Y. Li. (2020). "Clause vivification by unit propagation in CDCL SAT Solvers". *Artificial Intelligence*. 279(Feb.).

Liang, J. H., V. Ganesh, P. Poupart, and K. Czarnecki. (2016a). "Exponential recency weighted average branching heuristic for SAT solvers". In: *Proceedings of the 30th AAAI Conference on Artificial Intelligence (AAAI-16)*. 3434–3440.

Liang, J. H., V. Ganesh, P. Poupart, and K. Czarnecki. (2016b). "Learning Rate Based Branching Heuristic for SAT Solvers". In: *Proceedings of the 19th International Conference on Theory and Applications of Satisfiability Testing*. Ed. by N. Creignew and D. L. Berre. Vol. 9710. *Lecture Notes in Computer Science*. Springer. 123–140.

Liang, J. H., C. Oh, M. Mathew, C. Thomas, C. Li, and V. Ganesh. (2018). "Machine Learning-Based Restart Policy for CDCL SAT Solvers". In: *Proceedings of the 21st International Conference on Theory and Applications of Satisfiability Testing*. Ed. by O. Beyersdorff and C. M. Wintersteiger. Vol. 10929. *Lecture Notes in Computer Science*. Springer. 94–110.

Liang, J. H., H. G. P. Poupart, K. Czarnecki, and V. Ganesh. (2017). "An Empirical Study of Branching Heuristics Through the Lens of Global Learning Rate". In: *Proceedings of the 20th International Conference on Theory and Applications of Satisfiability Testing*. Ed. by S. Gaspers and T. Walsh. Vol. 10491. *Lecture Notes in Computer Science*. Springer. 119–135.

Lindauer, M., H. H. Hoos, F. Hutter, and T. Schaub. (2015). "Autofolio: an automatically configured algorithm selector". *Journal of Artificial Intelligence Research*. 53(1): 745–778.

Lonsing, F. and U. Egly. (2018). "Evaluating QBF Solvers: Quantifier Alternations Matter". In: *Proceedings of the 24th International Conference on Theory and Applications of Satisfiability Testing*. Ed. by J. Hooker. Vol. 11008. *Lecture Notes in Computer Science*. Springer. 276–294.

Loreggia, A., Y. Malitsky, H. Samulowitz, and V. Saraswat. (2016). "Deep Learning for Algorithm Portfolios". In: *Proceedings of the 30th AAAI Conference on Artificial Intelligence*. The AAAI Press. 1280–1286.

Luby, M., A. Sinclair, and D. Zuckerman. (1993). "Optimal speedup of Las Vegas algorithms". In: *Proceedings of the 2nd Israel Symposium on Theory and Computing Systems*. IEEE. 128–133.

Luenberger, D. (2003). *Linear and Nonlinear Programming*. 2nd Edition. Kluwer Academic Publishers.

Luo, M., C.-M. Li, F. Xiao, F. Manyà, and Z. Lü. (2017). "An Effective Learnt Clause Minimization Approach for CDCL SAT Solvers". In: *Proceedings of the 26th International Joint Conference on Artificial Intelligence (IJCAI)*. 703–711.

Lynce, I. and J. Marques-Silva. (2005). "Efficient data structures for backtrack search SAT solvers". *Annals of Mathematics and Artificial Intelligence*. 43(1–4): 137–152.

Lynce, I. and J. Marques-Silva. (2006). "Efficient Haplotype Inference with Boolean Satisfiability". In: *Proceedings of the 21st AAAI Conference on Artificial Intelligence*. Vol. 1. The AAAI Press. 104–109.

Malitsky, Y., A. Sabharwal, H. Samulowitz, and M. Sellmann. (2011). "Non-Model-Based Algorithm Portfolios for SAT". In: *Proceedings of the 14th International Conference on Theory and Applications of Satisfiability Testing*. Ed. by K. A. Sakallah and L. Simon. Vol. 6695. *Lecture Notes in Computer Science*. Springer. 369–370.

Malitsky, Y., A. Sabharwal, H. Samulowitz, and M. Sellmann. (2012). "Parallel SAT Solver Selection and Scheduling". In: *Proceedings of the 18th International Conference on Principles and Practice of Constraint Programming*. Ed. by M. Milano. Vol. 7514. *Lecture Notes in Computer Science*. Springer. 512–526.

Malitsky, Y., A. Sabharwal, H. Samulowitz, and M. Sellmann. (2013). "Algorithm Portfolios Based on Cost-Sensitive Hierarchical Clustering". In: *Proceedings of the 23rd International Joint Conference on Artifical Intelligence*. Ed. by F. Rossi. AAAI Press. 608–614.

Mangla, C., S. Holden, and L. Paulson. (2020). "Bayesian Optimization of Solver Parameters in CBMC". In: *Proceedings of the 18th International Workshop on Satisfiability Modulo Theories (SMT)*.

Marić, F. (2009). "Formalization, Implementation and Verification of SAT Solvers". *PhD thesis*. University of Belgrade.

Marques-Silva, J. (1999). "The Impact of Branching Heuristics in Propositional Satisfiability Algorithms". In: *Proceedings of the 9th Portugese Conference on Artificial Intelligence (EPIA)*. Ed. by P. Barahona and J. J. Alferes. Vol. 1695. *Lecture Notes in Computer Science*. Springer. 62–74.

Marques-Silva, J. (2008). "Practical Applications of Boolean Satisfiability". In: *Proceedings of the 9th International Workshop on Discrete Event Systems*. IEEE. 74–80.

Marques-Silva, J. and K. A. Sakallah. (1999). "GRASP: a search algorithm for propositional satisfiability". *IEEE Transactions on Computers*. 48(5): 506–521.

McCarthy, J. (1960). "Recursive functions of symbolic expressions and their computation by machine, Part I". *Communications of the ACM*. 3(4): 184–195.

McCune, W. (2003). "Otter 3.3 Reference Manual". *Tech. rep.* No. MCS-TM-263. 9700 South Cass Avenue, Argonne, IL 60439: Argonne National Laboratory.

McLaughlin, S. and F. Pfenning. (2008). “Imogen: Focusing the Polarized Inverse Method for Intuitionistic Propositional Logic”. In: *Proceedings of the 15th International Conference on Logic for Programming Artificial Intelligence and Reasoning (LPAR)*. Ed. by I. Cervesato, H. Veith, and A. Voronkov. Vol. 5330. *Lecture Notes in Computer Science*. Springer. 174–181.

Minton, S. (1996). “Automatically Configuring Constraint Satisfaction Programs: A Case Study”. *Constraints: An International Journal*. 1: 7–43.

Mitchell, M. (1998). *An Introduction to Genetic Algorithms*. The MIT Press.

Mitchell, T. (1997). *Machine Learning*. McGraw-Hill.

Mnih, V., K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, and D. Hassabis. (2015). “Human-level control through deep reinforcement learning”. *Nature*. 518(Feb.): 529–533.

Moskewicz, M. W., C. F. Madigan, Y. Zhao, L. Zhang, and S. Malik. (2001). “Chaff: engineering an efficient SAT solver”. In: *Proceedings of the 38th Design Automation Conference*. IEEE. 530–535.

Murphy, K. P. (2012). *Machine Learning: A Probabilistic Perspective*. MIT Press.

Nadel, A. and V. Ryvchin. (2018). “Chronological Backtracking”. In: *Proceedings of the 21st International Conference on Theory and Applications of Satisfiability Testing (SAT)*. Ed. by O. Beyersdorff and C. M. Wintersteiger. Vol. 10929. *Lecture Notes in Computer Science*. Springer. 111–121.

Narizzano, M., L. Pulina, and A. Tacchella. (2006). “The QBFEVAL Web Portal”. In: *Proceedings of the 10th European Workshop on Logics in Artificial Intelligence (JELIA)*. Ed. by M. Fisher, W. van der Hoek, B. Konev, and A. Lisitsa. Vol. 4160. *Lecture Notes in Computer Science*. Springer. 494–497.

Nejati, S., J. H. Liang, C. Gebotys, K. Czarnecki, and V. Ganesh. (2017). "Adaptive Restart and CEGAR-Based Solver for Inverting Cryptographic Hash Functions". In: *Proceedings of the 9th International Working Conference on Verified Software: Theories, Tools, and Experiments*. Ed. by A. Paskevich and T. Weis. Vol. 10712. *Lecture Notes in Computer Science*. Springer. 120–131.

Nieuwenhuis, R., A. Oliveras, and C. Tinelli. (2006). "Solving SAT and SAT Modulo Theories: From an abstract Davis-Putnam-Logemann-Loveland procedure to DPLL(T)". *Journal of the ACM*. 53(6): 937–977.

Nikolić, M., F. Marić, and P. Janičić. (2009). "Instance-Based Selection of Policies for SAT Solvers". In: *Proceedings of the 12th International Conference on Theory and Applications of Satisfiability Testing*. Ed. by O. Kullman. Vol. 5584. *Lecture Notes in Computer Science*. Springer. 326–340.

Nikolić, M., F. Marić, and P. Janičić. (2013). "Simple algorithm portfolio for SAT". *Artificial Intelligence Review*. 40: 457–465.

Nudelman, E., K. Leyton-Brown, H. H. Hoos, A. Devkar, and Y. Shoham. (2004). "Understanding Random SAT: Beyond the Clauses-to-Variables Ratio". In: *Proceedings of the 10th International Conference on Principles and Practice of Constraint Programming (CP)*. Ed. by M. Wallace. Vol. 3258. *Lecture Notes in Computer Science*. Springer. 438–452.

O'Mahony, E., E. Hebrard, A. Holland, C. Nugent, and B. O'Sullivan. (2008). "Using Case-based Reasoning in an Algorithm Portfolio for Constraint Solving". In: *Proceedings of the 19th Irish Conference on Artificial Intelligence and Cognitive Science*.

Oh, C. (2015). "Between SAT and UNSAT: The Fundamental Difference in CDCL SAT". In: *Proceedings of the 18th International Conference on Theory and Applications of Satisfiability Testing (SAT)*. Ed. by M. Heule and S. Weaver. Vol. 9340. *Lecture Notes in Computer Science*. Springer. 307–323.

Pearl, J. (1984). *Heuristics: Intelligent Search Strategies for Coimputer Problem Solving*. Addison-Wesley.

Peitl, T. and F. Slivovsky. (2017). “Dependency Learning for QBF”. In: *Proceedings of the 20th International Conference on Theory and Applications of Satisfiability Testing*. Ed. by S. Gaspers and T. Walsh. Vol. 10491. *Lecture Notes in Computer Science*. Springer. 298–313.

Petke, J., M. Harman, W. B. Langdon, and W. Weimer. (2014). “Using Genetic Improevement and Code Transplants to Specialise a C++ Program to a Problem Class”. In: *Proceedings of the 17th European Conference on Genetic Programming (EuroGP)*. Ed. by M. Nikolau, K. Krewiek, M. I. Heywood, M. Castelli, P. Garcia-Sánchez, J. J. Morello, V. M. R. Santos, and K. Sim. Vol. 8599. *Lecture Notes in Computer Science*. Springer. 137–149.

Petke, J., W. B. Langdon, and M. Harman. (2013). “Applying Genetic Improvement to MiniSAT”. In: *Proceedings of the 5th International Symposium on Search Based Software Engineering (SSBSE)*. Ed. by G. Ruhe and Y. Zhang. Vol. 8084. *Lecture Notes in Computer Science*. Springer. 257–262.

Pfahringer, B., H. Bensusan, and C. Giraud-Carrier. (2000). “Meta-Learning by Landmarking Various Learning Algorithms”. In: *Proceedings of the 17th International Conference on Machine Learning (ICML)*. Ed. by P. Langley. Morgan Kaufmann. 743–750.

Pierce, B. C. (2002). *Types and Programming Languages*. The MIT Press.

Pipatsrisawat, K. and A. Darwiche. (2007). “A Lightweight Component Caching Scheme for Satisfiability Solvers”. In: *Proceedings of the 10th International Conference on Theory and Applications of Satisfiability Tesing*. Ed. by J. Marques-Silva and K. A. Sakallah. Vol. 4501. *Lecture Notes in Computer Science*. 294–299.

Pisinger, D. and S. Ropke. (2010). “Large Neighborhood Search”. In: *Handbook of Metaheuristics*. Ed. by M. Gendreau and J.-Y. Potvin. Vol. 146. *International Series in Operations Research and Management Science*. Springer. 399–419.

Pulina, L. and A. Tacchella. (2007). "A Multi-engine Solver for Quantified Boolean Formulas". In: *Proceedings of the 13th International Conference on Principles and Practice of Constraint Programming*. Ed. by C. Bessière. Vol. 4741. *Lecture Notes in Computer Science*. Springer. 574–589.

Pulina, L. and A. Tacchella. (2009). "A self-adaptive multi-engine solver for quantified Boolean formulas". *Constraints*. 14: 80–116.

Quinlan, J. (1993). *C4.5: Programs for Machine Learning*. 1st Edition. Morgan Kaufmann.

Quinlan, J. R. (1986). "Induction of Decision Trees". *Machine Learning*. 1: 81–106.

Rabe, M. N. and S. A. Seshia. (2016). "Incremental Determinization". In: *Proceedings of the 19th International Conference on Theory and Applications of Satisfiability Testing (SAT)*. Ed. by N. Creignou and D. L. Berre. Vol. 9710. *Lecture Notes in Computer Science*. Springer. 375–392.

Raths, T., J. Otten, and C. Kreitz. (2007). "The ILTP Problem Library for Intuitionistic Logic". *Journal of Automated Reasoning*. 38: 261–271.

Rintanen, J. (1999). "Constructing Conditional Plans by a Theorem-Prover". *Journal of Artificial Intelligence Research*. 10: 323–352.

Rivest, R. L. (1987). "Learning Decision Lists". *Machine Learning*. 2(3): 229–246.

Russell, S. and P. Norvig. (2020). *Artificial Intelligence: A Modern Approach*. 4th ed. Pearson.

Saitta, L., A. Giordana, and A. Cornuéjols. (2011). *Phase Transitions in Machine Learning*. Cambridge University Press.

Samulowitz, H. and R. Memisevic. (2007). "Learning to Solve QBF". In: *Proceedings of the 22nd AAAI Conference on Artificial Intelligence*. The AAAI Press. 255–260.

Santos Silva, R. J. M. dos. (2019). "Machine learning of strategies for efficiently solving QBF with abstraction refinement". *MA thesis*. Instituto Superior Técnico, Universidade de Lisboa.

Schmee, J. and G. J. Hahn. (1979). "A Simple Method for Regression Analysis with Censored Data". *Technometrics*. 21(4): 417–432.

Sekiyama, T., A. Imanishi, and K. Suenaga. (2017). "Towards Proof Synthesis Guided by Neural Machine Translation for Intuitionistic Propositional Logic". arXiv: 1706.06462v1 [cs.PL].

Sekiyama, T. and K. Suenaga. (2018a). "Automated proof synthesis for propositional logic with deep neural networks". arXiv: 1805.11799v1 [cs.AI].

Sekiyama, T. and K. Suenaga. (2018b). "Automated Proof Synthesis for the Minimal Propositional Logic with Deep Neural Networks". In: *Proceedings of the 16th Asian Symposium on Programming Languages and Systems (APLAS)*. Ed. by S. Ryu. Vol. 11275. *Lecture Notes in Computer Science*. Springer. 309–328.

Selman, B., H. Kautz, and B. Cohen. (1996). "Local search strategies for satisfiability testing". In: *Cliques, Coloring and Satisfiability*. Ed. by D. S. Johnson and M. A. Trick. Vol. 26. *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*. American Mathematical Society. 521–532.

Selman, B., H. Levesque, and D. Mitchell. (1992). "A New Method for Solving Hard Satisfiability Problems". In: *Proceedings of the 10th National Conference on Artificial Intelligence*. Association for the Advancement of Artificial Intelligence. AAAI Press. 440–446.

Selsam, D. and N. Bjørner. (2019). "Guiding High-Performance SAT Solvers with Unsat-Core Predictions". In: *Proceedings of the 22nd International Conference on Theory and Applications of Satisfiability Testing (SAT)*. Ed. by M. Janota and I. Lynce. Vol. 11628. *Lecture Notes in Computer Science*. Springer. 336–353.

Selsam, D., M. Lamm, B. Bünz, P. Liang, L. de Moura, and D. L. Dill. (2019). "Learning a SAT Solver from Single-Bit Supervision". arXiv: 1802.03685 [cs.AI].

Shaw, P. (1998). "Using Constraint Programming and Local Search Methods to Solve Vehicle Routing Problems". In: *Proceedings of the 4th International Conference on Principles and Practice of Constraint Solving (CP)*. Ed. by M. Maher and J.-F. Puget. Vol. 1520. *Lecture Notes in Computer Science*. Springer. 417–431.

Shawe-Taylor, J. and N. Cristianini. (2000). *Support Vector Machines and Other Kernel-Based Learning Methods*. Cambridge University Press.

Shawe-Taylor, J. and N. Cristianini. (2004). *Kernel Methods for Pattern Analysis*. Cambridge University Press.

Silverthorn, B. (2012). “A Probabilistic Architecture for Algorithm Portfolios”. *PhD thesis*. The University of Texas at Austin.

Silverthorn, B. and R. Miikkulainen. (2010). “Latent Class Models for Algorithm Portfolio Methods”. In: *Proceedings of the 24th AAAI Conference on Artificial Intelligence*. Ed. by M. Fox and D. Poole. The AAAI Press. 167–172.

Singh, R., J. P. Near, V. Ganesh, and M. Rinard. (2009). “AvatarSAT: An Auto-tuning Boolean SAT Solver”. *Tech. rep.* No. MIT-CSAIL-TR-2009-039. MIT Computer Science and Artificial Intelligence Laboratory.

Soos, M., R. Kulkami, and K. S. Meel. (2019). “CrystalBall: Gazing in the Black Box of SAT Solving”. In: *Proceedings of the 22nd International Conference on Theory and Applications of Satisfiability Testing (SAT)*. Ed. by M. Janota and I. Lynce. Vol. 11628. *Lecture Notes in Computer Science*. Springer. 371–387.

Sörensson, N. and A. Biere. (2009). “Minimizing Learned Clauses”. In: *Proceedings of the 12th International Conference on Theory and Applications of Satisfiability Testing*. Ed. by O. Kullmann. Vol. 5584. *Lecture Notes in Computer Science*. Springer. 237–243.

Spears, W. M. (1996). “A NN Algorithm for Boolean Satisfiability Problems”. In: *Proceedings of the IEEE International Conference on Neural Networks*. 1121–1126.

Streeter, M. and D. Golovin. (2008). “An online algorithm for maximizing submodular functions”. In: *Proceedings of the 21st International Conference on Neural Information Processing Systems (NIPS)*. Ed. by D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou. Curran Associates Inc. 1577–1584.

Sutskever, I., O. Vinyals, and Q. V. Lee. (2014). “Sequence to Sequence Learning with Neural Networks”. In: *Advances in Neural Information Processing Systems (NIPS)*. Ed. by Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger. Vol. 2. 3104–3112.

Sutton, R. S. and A. G. Barto. (2018). *Reinforcement Learning: An Introduction*. 2nd Edition. MIT Press.

Tentrup, L. (2019). “CAQE and QuAbS: Abstraction based QBF solvers”. *Journal on Satisfiability, Boolean Modeling and Computation*. 11(1): 155–210.

Ting, K. M. (2002). “An instance-weighting method to induce cost-sensitive trees”. *IEEE Transactions on Knowledge and Data Engineering*. 14(3): 659–665.

Vaezipoor, P., G. Lederman, Y. Wu, R. Grosse, and F. Bacchus. (2020). “Learning Clause Deletion Heuristics with Reinforcement Learning”. In: *Proceedings of the Conference on Artificial Intelligence and Theorem Proving (AITP)*.

Vapnik, V. (2006). *Estimation of Dependencies based on Empirical Data*. Springer.

Vishwanathan, S. V. N., N. N. Schraudolph, R. Kondor, and K. M. Borgwardt. (2010). “Graph Kernels”. *Journal of Machine Learning Research*. 11: 1201–1242.

Wainberg, M., B. Alipanahi, and B. J. Frey. (2016). “Are Random Forests Truly the Best Classifiers?” *Journal of Machine Learning Research*. 17(1–5).

Wainer, J. and P. Fonseca. (2021). “How to tune the RBF SVM hyperparameters? An empirical evaluation of 18 search algorithms”. *Artificial Intelligence Review*. 54: 4771–4797.

Wang, P.-W., P. L. Donti, B. Wilder, and Z. Kolter. (2019). “SATNet: Bridging deep learning and logical reasoning using a differentiable satisfiability solver”. In: *Proceedings of the 36th International Conference on Machine Learning*. Ed. by K. Chaudhuri and R. Salakhutdinov. Vol. 97. *Proceedings of Machine Learning Research*. 6545–6554.

Wetzler, N., M. J. H. Heule, and W. J. H. Jr. (2014). “DRAT-trim: Efficient Checking and Trimming Using Expressive Clausal Proofs”. In: *Proceedings of the 17th International Conference on Theory and Applications of Satisfiability Testing (SAT)*. Ed. by C. Sinz and U. Egly. Vol. 8561. *Lecture Notes in Computer Science*. 422–429.

Williams, R. J. (1992). “Simple statistical gradient-following algorithms for connectionist reinforcement learning”. *Machine Learning*. 8(3–4): 229–256.

Wos, L. (1964). "The Unit Preference Strategy in Theorem Proving". In: *Proceedings of the Fall Joint Computer Conference (AFIPS)*. Association for Computing Machinery. 615–622.

Wu, Z., S. Pan, F. Chen, G. Long, C. Zhang, and P. S. Yu. (2019). "A Comprehensive Survey on Graph Neural Networks". arXiv: 1901.00596v4.

Xu, L., H. H. Hoos, and K. Leyton-Brown. (2007). "Hierarchical Hardness Models for SAT". In: *Proceedings of the 13th International Conference on Principles and Practice of Constraint Programming*. Ed. by C. Bessière. Vol. 4741. *Lecture Notes in Computer Science*. Springer. 696–711.

Xu, L., F. Hutter, H. Hoos, and K. Leyton-Brown. (2012a). "Evaluating Component Solver Contributions to Portfolio-Based Algorithm Selectors". In: *Proceedings of the 15th International Conference on Theory and Applications of Satisfiability Testing*. Ed. by A. Cimatti and R. Sebastiani. Vol. 7317. *Lecture Notes in Computer Science*. Springer. 228–241.

Xu, L., F. Hutter, H. H. Hoos, and K. Leyton-Brown. (2008). "SATzilla: Portfolio-based Algorithm Selection for SAT". *Journal of Artificial Intelligence Research*. 32: 565–606.

Xu, L., F. Hutter, H. H. Hoos, and K. Leyton-Brown. (2009). "SATzilla2009: an Automatic Algorithm Portfolio for SAT". In: *SAT 2009 competative events booklet*. 53–55.

Xu, L., F. Hutter, H. Hughes, and K. Leyton-Brown. (2012b). "Features for SAT". *Tech. rep.* University of British Columbia. URL: http://www.cs.ubc.ca/labs/beta/Projects/SATzilla/.

Xu, L., F. Hutter, J. Shen, H. H. Hoos, and K. Leyton-Brown. (2012c). "SATzilla2012: Improved Algorithm Selection Based on Cost-sensitive Classification Models". In: *Proceedings of SAT Challenge 2012: Solver and Benchmark Descriptions*. Ed. by A. Balint, A. Belov, D. Diepold, S. Gerber, M. Järvisalo, and C. Sinz. *Department of Computer Science Report Series B*. No. B-2012-2. Department of Computer Science, University of Helsinki. 57–58.

Yang, Z., F. Wang, Z. Chen, G. Wei, and T. Rompf. (2019). "Graph Neural Reasoning for 2-Quantified Boolean Formula Solvers". arXiv: 1904.12084v1 [cs.AI].

Yolcu, E. and B. Póczos. (2019). "Learning Local Search Heuristics for Boolean Satisfiability". In: *Proceedings of the 32nd Conference on Neural Information Processing Systems (NeurIPS)*. Ed. by H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché Buc, E. Fox, and R. Garnett. 7992–8003.

Yun, X. and S. L. Epstein. (2012). "Learning Algorithm Portfolios for Parallel Execution". In: *Proceedings of the 6th International Conference on Learning and Intelligent Optimization (LION)*. Ed. by Y. Hamadi and M. Schoenauer. Vol. 7219. *Lecture Notes in Computer Science*. Springer. 323–338.

Zhang, L., C. F. Madigan, M. H. Moskewicz, and S. Malik. (2001). "Efficient Conflict Driven Learning in a Boolean Satisfiability Solver". In: *Proceedings of the IEEE/ACM International Conference on Computer Aided Design*. 279–285.