ISSN 2186-7437

NII Shonan Meeting Report

No. 238

Model Counting: Theory meets Practice

Pierre Marquis (Université d'Artois, France)
Kenji Hashimoto (Kagawa University, Japan)
Johannes K. Fichte (Linköping University, Sweden)

February 2–6, 2026

National Institute of Informatics
2-1-2 Hitotsubashi, Chiyoda-Ku, Tokyo, Japan

Abstract

Propositional model counting (#SAT), asks to output the number of models of a propositional formula over the set of variables occurring in it. The #SAT problem is canonical for the complexity class #P and we can employ counting to solve any problem located on the Polynomial Hierarchy (PH). Over the last years, vast applications for counting have been identified and numerous solvers (aka counters) have been developed based on approximate or exact techniques. Especially, model counting is a key to the computation of exact probabilities in various fields. Very recent development shows progress in identifying and avoiding bugs in counters, enabling applications that require reliable and precise results. Despite the increasing practical success, theoretical research on quantitative solving and limitations falls short. Due to extremely diverse applications and numerous solving techniques, people interested in model counting come from various communities (including theoretical computer science, logic, and AI) and there have been surprisingly few interactions between these communities around model counting. We believe that there is an unexplored potential for reaching non-trivial advances in the state-of-the-art quantitative solving and theoretical understanding, especially under the light that progress in effective model counting yields substantial theoretical questions, applications may require new theoretical ideas for solving and stronger formalisms beyond propositional model counting. In this seminar, researchers came together to cross-fertilize improvements from diverse solving techniques, improving theoretical understanding of model counting, and significantly contribute to community building for practical applications.

Background and Introduction

The propositional satisfiability problem (SAT) asks to decide whether a given propositional formula is satisfiable, i.e., there exists an assignment to the variables of the formula such that the formula evaluates to true under the assignment [8]. A much harder problem is *propositional model counting* [Ch.23,24][8],[2], also known as *number SAT* or #SAT, instead of just asking for existence of a satisfying assignment it requires to output the number of total assignments to the variables of the formula, also known as *models*. The #SAT problem is canonical for the complexity class #P [49, 42, 2] and we can employ counting to solve any problem located on the Polynomial Hierarchy (PH) by an immediate consequence of Toda's Theorem [48]. Interestingly, #SAT remains computationally very hard on several classes of propositional formulas, where the SAT problem becomes tractable, such as 2-CNFs [49, 15]. However, very recent complexity results present a more fine-grained picture under more strict reductions for 2-CNFs [4, 5].

Despite the computational hardness, the field has seen considerable advances in recent years and highly efficient solvers emerged, capable of solving larger problems each year [13, 40, 33, 34, 43, 19]. These solvers have immediate use in various domains, such as quantitative questions in formal verification of hardware, software, security protocols, combinatorial optimization, computational mathematics, or analysing quantum circuits. Over the last few years, various applications can be witnessed in the literature [20, 27, 51, 21, 44, 3, 46, 35, 47, 23, 38] and will impact areas such as probabilistic reasoning [18, 36, 22], parameter learning in neuro-symbolic reasoning [37, 45, 50], or computational biology. An annual event, which evaluates solvers and is organized by the community since 2020 [25, 24, 28], provides a snapshot of the current state-of-the-art in model counting.

Today's, *#SAT solving techniques* are far more sophisticated than simply enumerating solutions, for example, solvers employ (i) component-caching, (ii) knowledge compilation, (iii) dynamic programming, and (iv) approximate counting. These techniques appear to be inherently different, but share many similarities. Recent articles [39, 30, 22] identify connections between structure from a theoretical perspective and solving approaches. Over the last years, approximate solving showed notable progress and dominated the perception from the outside. However, the last years demonstrated significant practical improvements in exact methods and component-caching-based solving [31, 32] and knowledge compilation [34] often outperform approximate techniques. *Knowledge compilation* (KC), which was pioneered almost three decades ago [29, 16, 17], is of high practical interest as it enables repeated fast counting by transforming an input formula into an equivalent (potentially large) formula on which counting is solvable in polynomial-time. Beyond counting, KC is of high interest in fields such as knowledge representation, constraint satisfaction, algorithms, complexity theory, machine learning, and databases.

Verifiable #SAT-solving recently emerged to identify and avoid bugs, which are still quite prevalent in today's solvers. The techniques work by either emitting a trace during solving [26] in a proof system focusing on simple constructs or showing equivalences [10, 9, 14] between the input formula and the resulting compiled formula. Then, the trace can be mathematically verified by a program, which in turn can be audited or formally verified. Interestingly, attempts for trustworthy solvers also yield new mathematical proof systems for counting.

The relation between proofs and solving is important as (i) proof systems can model aspects of solving, which allows for analyzing limitations of solvers by mathematical methods of proof complexity; and (ii) we can construct stronger proof systems, which enable to certify very sophisticated and complex solving techniques. Very recent work initiated the study into proof complexity of counting proof systems [6].

The need for (i) increasingly efficient, correct, and reliable solvers and (ii) robust theoretical understanding of limitations and capabilities of solving techniques is consistently on the rise, as (iii) more and harder problems need to be solved in various contexts. Although some of the research communities focus on the development of increasingly effective solving, theory, and modeling, there have been surprisingly few interactions between the communities when it comes to quantitative aspects. There remains significant unexplored potential for reaching non-trivial advances in the state-of-the-art quantitative solving. The seminar connected aspects (i)–(iii) and gathered leading researchers to pave the way for a fruitful cross-fertilization between the topics, from theory to practice and back.

Overview of the Meeting

The meeting brought together researchers working on the theory and practice of model counting, with contributions spanning counting complexity, practical #SAT solving, knowledge compilation, proof complexity, approximate counting, enumeration, planning, and applications of counting to Large Language Models (LLMs), neurosymbolic AI, quantum computing, and trustworthy AI. We saw numerous talks combining tutorials, focused research talks, and application-oriented presentations. We structured the meeting to encourage exchange beyond the formal talks, with short open-problem sessions, group discussions, and informal interactions. These discussions led to several concrete follow-up ideas, including incremental counting for plan-space navigation, new links between compilation formalisms, approximation questions, and further collaboration at the interface of model counting, quantum computing, and trustworthy AI systems. Overall, we saw three main topics:

(i) Cross-fertilizing improvements from diverse solving techniques: Today's best-performing solvers show diverging behavior depending on the application. Approximate solving behaves well on projected instances, is often sufficiently precise if run-time is negligible, and provides rough estimations on large instances. Exact solving such as component-caching and knowledge compilation is often faster, more precise, and enables other applications. Dynamic programming can easily be parallelized. Since the underlying techniques of the present solving approaches are quite orthogonal, we discussed questions such as extending solving techniques, benefiting from simplification techniques, and stronger solving approaches.

(ii) Improving theoretical understanding: While KC and its applications, in particular limitations and capabilities, have been studied extensively [17], many aspects of quantitative solving have been neglected with focus of theoretical research on decision and optimization problems and hardness results and algorithmic limitations. During the seminar, we revived theoretical questions on quantitative solving and discussed whether proof complexity can provide insights into quantitative problem solving, limitations in the quantitative show, and whether we can identify new techniques in KC to circumvent existing limitations.

(iii) Community building for practical applications: Improvements in #SAT-solving spawned many new applications in AI and natural sciences that do not reduce to decision problems. While an annual solver evaluation provides a platform for building tools and generating visibility outside of the immediate community, its focus has largely been on computational challenges. We discussed whether we can provide robust and easy-to-use solvers for a larger community, provide better insights for theoretical research, and what extensions are necessary for modern applications.

Overview of Talks

Short Tutorial: Basics in Counting Complexity

Heribert Vollmer, Leibniz Universität Hannover

We review different facets of studying the computational complexity of counting problems that have been proposed in the last 27 years. We consider complexity classes defined by counting paths of nondeterministic Turing machines running in polynomial time or logarithmic space. We consider Seinosuke Toda's predicate based approach, used the define counting analogues of the polynomial-time hierarchy. The predicate framework is also studied in the context of logarithmic space machines with a regular input tape plus a read-only input tape. For Boolean circuits, we consider classes defined by counting proof trees for circuits, thus obtaining arithmetic analogues of the classes AC^0, NC^1, SAC^1 and others. Finally, we also consider logically defined counting classes, defined by counting assignments to free second-order variables in formulas which are otherwise of first order.

We consider different types of reducibilities that are important for counting classes, and complete problems.

An occurring observation is that the quantifier sequence $\# \cdot \forall$ is very powerful.

Practical Model Counting

Jean-Marie Lagniez, Centre de Recherche en Informatique de Lens, Artois University, CNRS

This talk covers the state of the art in Exact Model Counting (#SAT). We move beyond basic DPLL algorithms to explore the " Holy Trinity" of modern counters: CDCL, Component Decomposition, and Caching. The presentation highlights the subtle implementation pitfalls of these systems, specifically the Cache Poisoning phenomenon caused by the interaction of learned clauses and local components. We also review cutting-edge features such as Probabilistic Component Caching and dynamic branching heuristics that allow modern solvers to function as efficient Knowledge Compilers.

Model Counting at ESA

Max Bannach, European Space Agency

This talk presents three application areas in which the AI and Data Science Section at the European Space Agency is exploring model counting as a potential enabling technology. First, we describe ongoing work on the use of model counting techniques for the simulation and verification of quantum circuits. Second, we consider the orbital reliability problem, a variant of the classical graph reliability problem in which nodes evolve over time according to Keplerian dynamics, introducing additional temporal and geometric constraints. Finally, we discuss the role of formal verification for neural networks in space applications and outline how model counting can provide a principled first step towards improving the interpretability and trustworthiness of data-driven models deployed in safety-critical settings.

Weighted Model Counting Problems in the Context of LLMs

Guy Van den Broeck, UCLA

This talk explores the role of weighted model counting and probabilistic circuits in addressing key challenges in large language models (LLMs), presenting applications in constrained text generation and alignment. Using Bayes' rule and tractable probabilistic circuits as "digital twins" of LLMs, we enable efficient reasoning about future tokens to satisfy hard constraints (Ctrl-G) or optimize soft attributes like toxicity (TRACE). The talk also addresses fundamental theoretical questions: characterizing when probabilistic circuits can be efficiently multiplied, and which circuit representations can capture all families of distributions that support efficient marginalization.

KLay: Accelerating Arithmetic Circuits for Neurosymbolic AI

Pedro Zuidberg Dos Martires, Örebro University

A popular approach to neurosymbolic AI involves mapping logic formulas to arithmetic circuits (computation graphs consisting of sums and products) and passing the outputs of a neural network through these circuits. This approach enforces symbolic constraints onto a neural network in a principled and end-to-end differentiable way. Unfortunately, arithmetic circuits are challenging to run on modern tensor accelerators as they exhibit a high degree of irregular sparsity. To address this limitation, we introduce knowledge layers (KLay), a new data structure to represent arithmetic circuits that can be efficiently parallelized on GPUs. Moreover, we contribute two algorithms used in the translation of traditional circuit representations to KLay and a further algorithm that exploits parallelization opportunities during circuit evaluations. We empirically show that KLay achieves speedups of multiple orders of magnitude over the state of the art, thereby paving the way towards scaling neurosymbolic AI to larger real-world applications.

Tutorial: Proof Complexity and Model Counting

Olaf Beyersdorff, Friedrich Schiller University Jena, Germany

We survey the area of proof complexity of model counting. Several proof systems for model counting have been suggested since 2019, most of which are inspired by solving approaches. The survey explains the different systems, static and line based, and discusses the simulation order of the systems and known separations. Many proof systems are based on Decision-DNNFs, which are suitably annotated. We also discuss how these proof systems relate to state-of-the-art #SAT solving approaches. In the end, we discuss future directions for the field.

Proof Systems That Tightly Characterise Model Counting Algorithms

Tim Hoffmann, Friedrich Schiller University Jena

Several proof systems for model counting have been introduced in recent years, mainly in an attempt to model #SAT solving and to allow proof logging of solvers. We reexamine these different approaches and show that: (i) with moderate adaptations, the conceptually quite different proof models of the dynamic system MICE and the static system of annotated Decision-DNNFs are equivalent and (ii) they tightly characterise state-of-the-art #SAT solving. Thus, these proof systems provide a precise and robust proof-theoretic underpinning of current model counting. We also propose new strengthenings of these proof systems that might lead to stronger model counters.

Approximate Model Counting (A Short Tutorial)

Kuldeep S. Meel, Georgia Institute of Technology

Model counting is the problem of computing the number of objects that satisfy a given specification, where the objects are described implicitly via formulas, automata, or combinatorial structures. While membership questions are often tractable, exact counting is typically #P-hard even when the underlying decision problem is in P. Consequently, the focus has shifted to approximate counting via randomized algorithms, with the goal of designing schemes that provably output a value within a $(1 \pm \varepsilon)$ factor of the true count with high probability, where the running time is polynomial in the input size and $1/\varepsilon$. This talk surveys the fundamental ideas that make such approximation possible, organized around four recurring paradigms that illuminate both the theoretical foundations and practical realities of approximate counting.

The first paradigm employs Monte Carlo methods that construct unbiased estimators whose expectation equals the desired count, with the central challenge being to ensure the estimator's expectation has polynomial lower bounds relative to the true count, enabling concentration via Chernoff bounds. The second paradigm exploits the inter-reducibility between counting and sampling via self-reducibility and telescoping products, with applications ranging from the permanent to linear extensions and volume estimation. The third paradigm combines sampling with structural decompositions in automata and circuit representations, exemplified by recent breakthroughs in NFA counting via suffix-growing strategies and extensions to DNNF model counting. The fourth paradigm leverages hashing-based techniques to partition the solution space into cells, enabling estimation with polynomial SAT oracle calls and leading to the first scalable practical tools with rigorous guarantees. We illustrate these ideas through canonical problems including DNF counting, knapsack counting, counting matchings, NFA counting, and CNF model counting. Throughout, we highlight the tension between worst-case theoretical analysis and empirical performance, examining where theory and practice converge and where significant gaps remain.

Semantic Structure for SDD Compilation

Stefan Szeider, TU Wien

Existing vtree construction methods treat CNF formulas as flat hypergraphs. This approach does not distinguish between problem variables and the auxiliaries that encoders introduce. The basic idea for our approach is to first build a skeleton over semantic variables organized by problem dimensions, and then to attach each auxiliary at the lowest common ancestor of its associated semantic variables. These vtrees can be refined through dynamic minimization during bottom-up compilation. On graph coloring over grids, our method compiles 12×12 instances while PySDD's search times out at 9×9. The resulting SDDs are 95–99.98% smaller than c2d's d-DNNFs, even though SDD is a stricter representation. This gap grows with problem size. We find that semantic vtrees work best for local constraints with many auxiliaries. Hypergraph partitioning wins when constraints span entire dimensions.

Joint work with Alexis de Colnet.

Message Passing algorithms for computation of error metrics in probabilistic and approximate circuits

Vinita Vasudevan, Indian Institute of Technology Madras

Many error-resilient applications, especially those in image and speech processing and wireless communications use approximate computing to trade accuracy for energy efficiency. The two techniques commonly used are (a) Scaling down the supply voltage and (b) Simplification/removal of portions of the circuit. At very low supply voltages we get significant energy savings, but the gate behaviour is probabilistic. Simplification gives power savings by changing the functionality of the circuit. These approximate circuits need to be implemented with an overall constraint on the output error. The system to be analysed consists of the fault-free and faulty circuit along with an error miter. The construction of the miter depends on the error metric used. Several metrics have been proposed in the literature. Computation of nearly all error metrics is a #SAT problem, with the error miter itself being a significant source of complexity. BDDs have been used to compute the error, but have shown limited scalability.

We have attempted two different approaches, both of which are based on message-passing algorithms. The first approach is based on incremental build and approximation of clique trees (IBIA). The second approach is a partition-merge and message-passing technique, in which we relax memory constraints and try to push the boundaries of circuits for which we can obtain the exact error for a variety of error metrics. In this talk, I will outline the algorithms we have proposed, some recent results and the successes and limitations of our methods. It represents joint work with Shivani Bathla, S.Ramprasath, MGKS Charan and Sibi Siddharthan.

The material on IBIA is published in TMLR, Sept 2023 and Neurips 2023. The material on probabilistic circuits is published in IEEE Transactions on VLSI, 2023. The work on approximate circuits is as yet unpublished and is available on ArXiv [41] at `https://arxiv.org/abs/2411.10037`.

An Explainable Planning Framework using Model Counting

Daniel Gnad, Heidelberg University

For many planning applications it is not sufficient to provide a single plan that solves a planning task, even if the solver guarantees that the solution is optimal with respect to some quality metric. To be useful in practice, it is important that practitioners are able to obtain some justification for why the solution returned by an automated planner has been selected. Ideally, there is the possibility to modify the given solution to adjust it further by the application expert, since often not all quality criteria can be directly encoded in the task. We present an approach based on model counting that allows to (1) obtain counterfactual explanations that provide the user with additional information on how the returned solution differs from alternatives, and (2) an interactive tool that enables iterative plan-refinement via systematic navigation of the solution space of the planning task.

Since exact counting, being #P-complete for bounded planning, is challenging for many non-trivial planning problems, we suggest an approximation based on facet reasoning, which lies in NP, scaling to much larger instances and still allowing to compute action significance scores, which are at the core of the plan-space navigation.

Efficient Volume Computation for SMT Formulas

Arijit Shaw, Chennai Mathematical Institute

Satisfiability Modulo Theory (SMT) has recently emerged as a powerful tool for solving various automated reasoning problems across diverse domains. Unlike traditional satisfiability methods confined to Boolean variables, SMT can reason on real-life variables like bitvectors, integers, and reals. A natural extension in this context is to ask quantitative questions. One such query in the SMT theory of Linear Real Arithmetic (LRA) is computing the volume of the entire satisfiable region defined by SMT formulas. This problem is important in solving different quantitative verification queries in software verification, cyber-physical systems, and neural networks, to mention a few. We introduce ttc, an efficient algorithm that extends the capabilities of SMT solvers to volume computation. Our method decomposes the solution space of SMT Linear Real Arithmetic formulas into a union of overlapping convex polytopes, then computes their volumes and calculates their union. Our algorithm builds on recent developments in streaming-mode set unions, volume computation algorithms, and AllSAT techniques. Experimental evaluations demonstrate significant performance improvements over existing state-of-the-art approaches.

Sometimes Counting is not Enough: Why Model Enumeration Matters

Giuseppe Spallitta, Rice University

Model counting is a powerful abstraction for reasoning about large solution spaces, but in some settings, the structure of individual models becomes essential. This talk focuses on model enumeration as a complementary approach, emphasizing how it enables downstream tasks such as probabilistic inference. Key algorithmic aspects that arise in enumeration are discussed, including the role of preprocessing, structural simplification, and the cost of reasoning with partial models. Two applications illustrate this perspective: (*i*) structure-aware enumeration for WMI, where extracting the Boolean skeleton of the weight and using enumeration-friendly transformations reduces the number of integration regions and can reach an optimal subdivision; and (*ii*) theory-aware knowledge compilation, where enumeration is used to extract T-lemmas that eliminate theory-inconsistent assignments before OBDD/SDD compilation, enabling canonicity to the SMT level.

Complexity of Enumerating Satisfying Assignments

Heribert Vollmer, Leibniz Universität Hannover

We study the computational complexity of model enumeration for different syntactically defined classes of propositional formulas and quantified propositional formulas. First, we extend the well-known class DelayP, introduced by Johnson, Papadimitriou and Yannakakis, by oracles, and obtain an analogue of the polynomial-time hierarchy for enumeration problems. On the other hand, we define subclasses defined by Boolean circuits as enumerators.

Parameterized Enumeration

Arne Meier, Leibniz Universität Hannover

After reviewing the basics of parameterised complexity theory, we provide an introduction to parameterised enumeration problems. We then illustrate two key techniques—kernelisation and self-reducibility—from the perspective of parameterised enumeration. We demonstrate enum-kernels using the vertex-cover problem and self-reducibility using the MaxOnes-SAT problem.

Finally, we turn to parameterised *ordered* enumeration. First, we present a DelayFPT result with polynomial space for lexicographic orders and graph modification problems. Next, we introduce the concept of neighbourhood functions. Here, the notion of forbidden set characterisations are helpful to provide efficient DelayFPT enumeration, as illustrated by the cluster-editing problem, that adhere to a non-decreasing size order.

Finally, we conclude with recent refinements of enum-kernel notions and applications of the framework.

The Return of Symmetry-Driven Compilation Languages

Alexis de Colnet, Leiden University

The last model counting competition witnessed the arrival of #SAT solvers with caching strategies using symmetry detection. Since the model count is invariant under formula isomorphism, these solvers detect when a formula is isomorphic to another formula already encountered to avoid wasteful computations. In this talk, I revisit the knowledge compilation languages studied in the 2014 paper "Symmetry-Driven Decision Diagrams for Knowledge Compilation" by Bart, Koriche, Lagniez and Marquis, which I believe provide a nice framework to represent the traces of #SAT solvers with symmetry detection. This talk is meant to be a helpful reminder about these languages and their properties but also presents new results.

Advancing Quantum Computing with Model Counting

Alfons Laarman, Leiden University

Decades of research into satisfiability has yielded powerful solvers. Initial attempts to harness SAT and SMT solvers for quantum circuit analysis yielded some success, but these results were also limited to non-universal circuits or a few qubits. By switching from SAT/SMT to weighted model counting (WMC or #SAT), we recently enabled the use of state-of-the-art solvers for quantum circuit analysis, which comprises simulation, equivalence checking, Hoare-logic verification, and synthesis. An implementation of this method, called Quokka#, outcompetes other state-of-the-art approaches using an off-the-shelf #SAT solvers extended with support for negative and complex weights (Ganak and GPMC). Finally, we show that model counters can also be used to compute the partition function for quantum systems. Our implementation, DiracWMC, automatically encodes linear algebra problems in Dirac notation as WMC instances.

In the second part of the talk, we study knowledge compilation approaches for quantum circuit analysis. Hitherto, decision diagrams were extensively used here. However, we prove that decision diagrams cannot succinctly represent states with high entanglement, in particular, the so-called stabilizer states. Stabilizer states are ubiquitous in quantum computing, and the circuits generating them are (efficiently) classically simulatable using the so-called stabilizer formalism, making this limitation all the more concerning. To alleviate this bottleneck, we introduce the Local Invertible Map Decision Diagram (LIMDD), which offers exponential improvements in succinctness over the combination of the stabilizer formalism and existing decision diagrams. We then provide a first knowledge compilation map for quantum information, comparing various decision diagrams against tensor networks and Boltzmann machines.

Our results show that existing model counting approaches have strong potential for quantum computing and physics.

A Compositional Atlas for Algebraic Circuits

YooJung Choi, Arizona State University

Circuits based on sum-product structure have become a ubiquitous representation to compactly encode knowledge, from Boolean functions to probability distributions. By imposing constraints on the structure of such circuits, certain inference queries become tractable, such as model counting and most probable configuration. This talk presents how we can analyze various queries as compositions of basic operators to derive tractability conditions. In particular, we take an algebraic perspective for compositional inference and show that a large class of queries—including marginal MAP, probabilistic answer set programming inference, and causal backdoor adjustment—correspond to a combination of basic operators over semirings: aggregation, product, and elementwise mapping. Using this framework, we uncover simple and general sufficient conditions for tractable composition of these operators, in terms of circuit properties (e.g., marginal determinism, compatibility) and conditions on the elementwise mappings. Applying our analysis, we derive novel tractability conditions for many such compositional queries. Our results unify tractability conditions for existing problems on circuits, while providing a blueprint for analyzing novel compositional inference queries.

Extension Complexity and the Convex Hull of Boolean Functions

Florent Capelli, Université d'Artois

One can naturally map a Boolean function f over n variables as a convex set of $[0,1]^n$ as the convex hull of its models, seen as points in $\{0,1\}^n$. By definition, the extreme points of this polytope are the models of f. Hence, we know that the optimal value of any linear function over $x_1, \ldots, x_n$ will be reached on an extreme point of this polytope, that is, at a model of f, enabling the use of linear solver for this kind of tasks. Unfortunately, the description of f as a linear program can be of exponential size. From this point of view, it becomes natural to look at polynomial sized representation of the convex hull of f in terms of linear programs. A natural tool for this is to use extended formulation: an extended formulation for a polytope Q is a polytope P in a larger space such that the projection of P onto a smaller space matches Q. Ideally, P can be described with a smaller number of linear constraint than Q. The smallest way of describing Q this way is called the extension complexity of Q. We can then naturally define the extension complexity of f to be the extension complexity of its convex hull.

In this talk, we exhibit an interesting connection between knowledge compilation and extension complexity of f by showing that the Boolean function computed by a DNNF C has extension complexity $O(|C|)$. We will start this talk by using the example of OBDD to build in some natural intuition on this connection before generalizing to DNNF. We will then explore the consequences and applications of this connection.

The work presented here is adpated from Section 6 of the still unpublished paper "A Knowledge Compilation Take on Binary Polynomial Optimization" in collaboration with Alberto Del Pia and Silvia Di Gregorio [12], which is available at `https://arxiv.org/abs/2311.00149`.

The Model Counting Competition

Johannes K. Fichte, Linköping University

The Model Counting Competition was initiated by the community in 2019, with its first iteration in 2020. It aims to deepen the relationship between the latest theoretical and practical developments in implementation and their applications in propositional model counting. The competition is inspired by the success of the SAT and automated reasoning community, with persistent efforts to improve the performance and robustness of solvers in evaluations and challenges. In the competition, we challenge the community to improve solvers, identify benchmarks, and facilitate the exchange of novel ideas. Beyond the interest group, we make a large audience of researchers aware of counting techniques and their use, particularly in cases where enumerating models is expensive and unnecessary, yet is still a common approach. In this talk, we present an overview of design decisions, their execution, and their outcomes. The main problems centered on the model counting problem (MC), the weighted model counting problem (WMC), projected model counting (PMC), and their combinations (PWMC).

See: `https://mccompetition.org/`

Auditable Algorithms for Approximate Model Counting

Supratik Chakraborty, IIT Bombay

Model counting, or counting the satisfying assignments of a Boolean formula, is a problem with diverse applications . Given #P-hardness of the problem, developing algorithms for approximate counting is important from a practical perspective. Building on the practical success of SAT-solvers, the focus has recently shifted from theory to practical implementations of approximate counting algorithms. This has brought to focus new challenges, such as the design of auditable approximate counters that not only provide an approximation of the model count, but also a certificate that a verifier with limited computational power can use to check if the count is indeed within the promised bounds of approximation. In this talk, we discuss certificates for deterministic approximate model counters (DAC). We start by examining the best-known DAC algorithm (due to Stockmeyer) that uses polynomially many calls to a Σ_2^P oracle. We show that this can be audited via a Σ_2^P oracle with the query constructed over $\mathcal{O}(n^2 \log^2 n)$ variables, where the original formula has n variables. Since n is often large, we ask if the count of variables in the certificate can be reduced – a crucial question from an implementation perspective. We show that this is indeed possible with a tradeoff in the counting algorithm's complexity. Specifically, we develop new deterministic approximate counting algorithms that invoke a Σ_3^P oracle, but can be certified using a Σ_2^P oracle using certificates on far fewer variables: our final algorithm uses only $\mathcal{O}(n \log n)$ variables. Our study demonstrates that one can simplify auditing significantly if we allow the counting algorithm to access a slightly more powerful oracle. This shows for the first time how audit complexity can be traded for complexity of approximate counting.

This talk is based on joint work with Kuldeep Meel and S. Akshay, and published at AAAI 2024

List of Participants

- S. Akshay (Indian Institute of Technology Bombay, India) [dblp]
- Max Bannach (European Space Agency, The Netherlands) [dblp]
- Olaf Beyersdorff (Friedrich Schiller University Jena, Germany) [dblp]
- Florent Capelli (University of Artois, France) [dblp]
- Supratik Chakraborty (Indian Institute of Technology Bombay, India) [dblp]
- Arthur Choi (Kennesaw State University, United States) [dblp]
- YooJung Choi (Arizona State University, United States) [dblp]
- Alexis de Colnet (Leiden University, The Netherlands) [dblp]
- Arnaud Durand (University Paris Cite, France) [dblp]
- Johannes Klaus Fichte (Linköping University, Sweden) [dblp]
- Daniel Gnad (Heidelberg University, Germany) [dblp]
- Kenji Hashimoto (Kagawa University, Japan) [dblp]
- Tim Hoffmann (Friedrich Schiller University Jena, Germany) [dblp]
- Matti Järvisalo (University of Helsinki, Finland) [dblp]
- Jean-Marie Lagniez (University of Artois, France) [dblp]
- Kuldeep Meel (Georgia Institute of Technology, United States) [dblp]
- Arne Meier (Leibniz University Hannover, Germany) [dblp]
- Arijit Shaw (Chennai Mathematical Institute, India) [dblp]

- Giuseppe Spallitta (Rice University, United States) [dblp]
- Stefan Szeider (TU Wien, Austria) [dblp]
- Guy Van den Broeck (UCLA, United States) [dblp]
- Vinita Vasudevan (Indian Institute of Technology Madras, India) [dblp]
- Heribert Vollmer (Leibniz Universität Hannover, Germany) [dblp]
- Pedro Zuidberg Dos Martires (Örebro University, Sweden) [dblp]

Meeting Schedule

Check-in Day: February 1, 2026 (Sun)

- Welcome Banquet

Day 1: February 2, 2026 (Mon)

- 09:00-09:15
  Welcome by the Organizers (Room 208)
  by Kenji Hashimoto, Johannes Fichte
- 09:15-10:15
  Introduction by the Participants
- 10:45-11:15
  Short Tutorial: Basics in Counting Complexity
  by Heribert Vollmer
- 13:30-14:30
  Tutorial: Practical Knowledge Compilation
  by Jean-Marie Lagniez
- 14:30-15:00
  Model Counting at ESA (European Space Agency)
  by Max Bannach
- 16:00-16:30
  Weighted Model Counting Problems in the Context of LLMs
  by Guy Van den Broeck
- 16:30-17:00
  KLay: Accelerating Arithmetic Circuits for Neurosymbolic AI
  by Pedro Zuidberg Dos Martires
- 17:00-17:45
  Open Problem Session
  lead by Florent Capelli

Day 2: February 3, 2026 (Tue)

- 09:00-10:00
  Tutorial: Proof Complexity & Model Counting
  by Olaf Beyersdorff

- 10:30-11:00
  Proof Systems That Tightly Characterise Model Counting Algorithms
  by Tim Hoffmann
- 11:00-11:30
  Semantic Vtrees for SDD Compilation
  by Stefan Szeider
- 13:30-14:15
  Short Tutorial: Approximate Model Counting
  by Kuldeep Meel
- 14:15-14:45
  Message Passing Algorithms for Computation of Error in Probabilistic and Approximate Circuits
  by Vinita Vasudevan
- 16:00-16:30
  Counting and Planning
  by Daniel Gnad
- 16:30-17:00
  Counting Beyond #SAT: Quantification in SMT
  by Arijit Shaw
- 17:00-18:00
  Group Sessions

Day 3: February 4, 2026 (Wed)

- 09:00-09:30
  Sometimes Counting Is Not Enough: Why Model Enumeration Matters
  by Giuseppe Spallitta
- 09:30-10:15
  Efficient Enumeration on Circuits by Heribert Vollmer
- 10:45-11:30
  Parameterized Enumeration
  by Arne Meier
- 13:30-18:00
  Excursion to Kamakura Visiting Jomyoji and Hokokuji temple with Japanese Tea ceremony

Day 4: February 5, 2026 (Thu)

- 09:00-09:30
  The Return of Symmetry-Driven Compilation Languages
  by Alexis de Colnet
- 09:30-10:00
  Quantum Circuit Analysis
  by Alfons Laarman
- 10:45-11:30
  Groups Session

- 13:30-14:00
  A Compositional Atlas for Algebraic Circuits
  by YooJung Choi
- 14:00-14:30
  Extension Complexity and the Convex Hull of Boolean Functions
  by Florent Capelli
- 14:30-18:00
  Group Sessions

Day 5: February 6, 2026 (Fr)

- 09:00-09:30
  The Model Counting Competition & Open Discussions
  by Johannes Fichte
- 09:30-10:00
  Auditable Algorithms for Approximate Model Counting
  by Supratik Chakraborty
- 10:30-11:15
  Outcomes and Follow Ups
- 11:15-11:30
  Closing by Kenji Hashimoto and Johannes Fichte

Summary of Discussions

Supporting Repeated Iterative Counting for Plan-space Navigation

Jean-Marie Lagniez, Daniel Gnad

Currently, a practical limitation of the plan-space navigation implemented in PlanPilot is that the counter is started from scratch for every single counting query. This is despite the fact that the queries almost always differ only in a single variable, i.e., in every step of the navigation a single new action is either enforced or forbidden, which fixes the assignment to a single SAT variable. Jean-Marie suggested to not start the solver (d4) from scratch, but to keep it running, add the new assumption, and re-query the solver. Due to internal caching, this is very likely to tremendously speed up the navigation. The Python bindings currently under development will further simplify the integration of d4, avoiding the current interaction which is heavy on text I/O.

Open Complexity for Several Counting Problems in Planning

Arnaud Durand, Daniel Gnad

Problems currently open (to the best of my knowledge):

- Counting the number of unique loopless plans. A plan is loopless if it does not visit the same state more than once. Arnaud has a promising pointer to trail/acyclic semantics in the graph query language (GraphQL).

- Plans are sequences of actions, but often different sequences exist for the same (multi-)set of actions, which are just reorderings of each other. Counting the number of plans that differ in their underlying action (multi-)set seems to be harder than just counting all plans (which is #P). No concrete ideas yet, just the simple observation that in the worst case, a plan with n actions can be ordered in $n!$ different ways.

Comparing SDD with a New Generalization of OBDD

Florent Capelli, YooJung Choi, Guy Van den Broeck

Model counting can often be done by first compiling the CNF formula into a Boolean circuit for which we can extract the number of models. One approach popular approach for building such circuit is to use a bottom-up strategy: first compile a small circuit for every clause of the formula and then iteratively combine these circuits together into a larger circuit computing the conjunction of the previous ones. This approach has been used to compile so-called Ordered Binary Decision Diagrams (OBDD) and Sentential Decision Diagrams. Stefan Mengel and Florent Capelli recently started studying a natural generalization of OBDDs to treelike structures called Tree Decision Diagrams (TDD). TDDs have many interesting properties. In particular, every TDD has a canonical minimal representation that can be computed from it by contracting its gate. It allows to keep the circuit as small as possible during the compilation phase. It was not clear however how these circuits compare in terms of size with SDDs. Florent, YooJung and Guy spent some time discussing this question and finally managed to prove that every TDD can be encoded into a polynomial size SDD. This has led to a join submission at SAT'26 where the TDD data structure is studied in details, see [11] for a preprint. `https://arxiv.org/abs/2604.05537`

Linear Programs and FPRAS

Florent Capelli, Kuldeep Meel

Florent's talk has presented a way of encoding the models of Boolean functions as the extreme points of a polyhedron described by a small number of linear constraints. These constraints can be extracted from many representations of Boolean functions used in knowledge compilation. In other words, we could interpret some linear programs as succinct representations of Boolean functions, allowing to solve some tasks efficiently such as enumerating every model. However, the problem of counting the number of extreme points of such polyhedron, and hence of find the number of models of the underlying Boolean function, is #P-hard. Florent and Kuldeep discussed the possibility of designing an efficient approximation algorithm (FPRAS) in this case, which would allow to generalize recent results from Kuldeep Meel and Alexis Colnet about designing an FPRAS algorithm for DNNF circuits.

Increasing the Trustworthiness of AI Systems in Mission-Critical Subsystems

S Akshay, Max Bannach, Supratik Chakraborty

A major limitation of deploying AI onboard spacecraft is trustworthiness. Certifying black-box components for use in mission-critical subsystems remains difficult, if not impossible, under current verification standards. During the workshop, we discussed the use of model counting techniques to verify binarized neural networks and to estimate the reliability of satellite constellations with inter-satellite links. We also explored an alternative approach to improving the trustworthiness of AI systems based on automated auditing, which led to the identification of a potential future collaboration.

PDL Knowledge Compilation

Arnaud Durand, Arne Meier, Heribert Vollmer

Motivated by talks on knowledge compilation, we considered propositional Dependence Logic (PDL) which extends classical logic by incorporating dependence atoms, $\mathrm{dep}(x_1, \ldots, x_n, y)$, which explicitly model functional relationships between variables within a set of assignments. We discussed encoding of the PDL model checking problem (known to be NP-complete) into SAT. Furthermore, we pondered about circuit-based representations and DNNFs.

From Classical to Quantum Circuits, and back

Yoojung Choi, Florent Capelli, Jean-Marie Lagniez, Alfons Laarman

We explored the intersection of quantum computing and automated reasoning, looking for novel ways to apply model counting and related approaches to reason about quantum circuits. We identified new normal forms that allow efficient "squaring" of quantum circuits. Early investigations show that this yields dramatic improvements over previous approaches. Further, we discussed compositional methods for applying different optimization operations to quantum circuits. This led to new methods for quantum circuit synthesis, which circumvent the use of expansive oracles.

We also discussed a new canonical normal form for Boolean functions called tree decision diagram (TDD). Such results can easily be extended to pseudo-Boolean functions, and since quantum states are pseudo-Boolean functions, we foresee many open possibilities for cross-fertilization between the two fields.

After the workshop, Yoojung Choi visited Alfons Laarman's team in Leiden to explore these topics in more depth. This led to improved methods for incorporating stabilizer states [1] into model counting procedures.

Further directions for #SAT proof systems

Olaf Beyersdorff, Tim Hoffmann

Recently, we proposed a new proof system [7] and showed that it theoretically characterises state-of-the-art solvers such as D4. For this result we used an abstracted and simplified version of the underlying algorithms. As our work is purely theoretical, the exchange with practitioners provided valuable insights and perspectives. In particular, Jean-Marie Lagniez emphasised that pre- and in-processing techniques are absolutely crucial for the high performance that these solvers achieve. Therefore, we aim to extend our proof system such that it can also capture these pre- and in- processing steps. This may also lead to theoretical insights into why these techniques are so powerful.

Identified Issues and Future Directions

During the seminar, we identified three main challenges and future directions.

Strengthening the connection between theory and practice Clearly, there are several lines of theoretical and practical work, many of them developed in different fields such as computational complexity, databases, logic, and quantitative solving. While results would be of joint interest, closely related questions are often studied with different methods, terminologies, and venues, but also with slightly different focus, e.g., effective solving vs. highly generic hierarchies of hardness and efficiency, which are practically of limited relevance. A natural future direction is therefore to create more regular points of contact between these communities, for instance through focused workshops or follow-up meetings in formats such as FLoC workshops, Dagstuhl seminars, Banff workshops, or meetings at the Lorentz Center.

Broad use of counting techniques and systems for applications The meeting also made clear that numerous application areas (e.g., planning or quantum computing) could benefit from counting, but that current solvers, platforms, and workflows are still not sufficiently accessible to a broader group of users. In several emerging directions, stronger support for extended reasoning is required, including effective handling of richer algebraic domains such as negative and complex weights, as motivated for example by applications in quantum computing and reliability questions arising in future space missions. This points to the need for sustained investment into solver extensions, extensions of proof systems, and software infrastructures that support reasoning at larger scale, including parallel and incremental solving.

Stronger solving techniques Another central challenge concerns the relation between solver design, modeling choices, and empirical performance. Novel ideas for solving remain, unsurprisingly, of broad interest. But the meeting also highlighted the need for a more systematic understanding of which techniques work well on which classes of instances, and why. This includes stronger modeling methodologies, improved compilation-based and caching-based approaches, as well as a better theoretical account of the capabilities and limitations of current systems.

Finally, the participants identified during the meeting that there is a general lack of accessible introductory material on model counting and closely related directions such as enumeration. While the field has developed already in the

'80s and '90s, tutorials, surveys, and systematic entry points for students, non-specialists, and potential users of counting software remain limited. Several participants therefore saw a clear need for a joint community effort to develop survey-style material, introductory tutorials, and use- case-driven resources that make the main concepts, techniques, and tools of the area more widely accessible.

References

[1] Aaronson, S., and Gottesman, D. Improved simulation of stabilizer circuits. *Physical Review A—Atomic, Molecular, and Optical Physics 70*, 5 (2004), 052328.

[2] Bacchus, F., Dalmao, S., and Pitassi, T. Algorithms and complexity results for #SAT and Bayesian inference. In *Proceedings of the 44th IEEE Symposium on Foundations of Computer Science (FOCS'03)* (Cambridge, MA, USA, Oct. 2003), C. S. Chekuri and D. Micciancio, Eds., IEEE Computer Soc., pp. 340–351.

[3] Baluta, T., Chua, Z. L., Meel, K. S., and Saxena, P. Scalable quantitative verification for deep neural networks. In *Proceedings of the 43rd IEEE/ACM International Conference on Software Engineering (ICSE'21)* (Madrid, 2021), IEEE Computer Soc., pp. 312–323.

[4] Bannach, M., Demaine, E. D., Gomez, T., and Hecher, M. #P is sandwiched by one and two #2dnf calls: Is subtraction stronger than we thought? In *Proceedings of the 40th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS'25)* (2025), pp. 31–43.

[5] Bannach, M., Demaine, E. D., Gomez, T., and Hecher, M. A novel reduction from #sat to #2sat based on symmetry: Simply drop the large clauses. In *Proceedings 2026 SIAM Symposium on Simplicity in Algorithms (SOSA'26)* (2026), pp. 254–265.

[6] Beyersdorff, O., Fichte, J. K., Hecher, M., Hoffmann, T., and Kasche, K. The relative strength of #sat proof systems. In *Proceedings of the 27th International Conference on Theory and Applications of Satisfiability Testing (SAT'24)* (2024), S. Chakraborty and J. R. Jiang, Eds., LIPIcs, Schloss Dagstuhl - Leibniz-Zentrum für Informatik, pp. 5:1–5:19.

[7] Beyersdorff, O., Hoffmann, T., and Kasche, K. Proof systems that tightly characterise model counting algorithms. In *Proceedings of the 40th AAAI Conference on Artificial Intelligence (AAAI'26)* (2026), S. Koenig, C. Jenkins, and M. E. Taylor, Eds., AAAI Press, pp. 14184–14191.

[8] Biere, A., Heule, M., van Maaren, H., and Walsh, T., Eds. *Handbook of Satisfiability (2nd Edition).* Frontiers in Artificial Intelligence and Applications. IOS Press, Amsterdam, Netherlands, Feb. 2021.

[9] Bryant, R. E., Nawrocki, W., Avigad, J., and Heule, M. J. H. Certified Knowledge Compilation with Application to Verified Model Counting. In *Proceedings of the 26th International Conference on Theory and Applications of Satisfiability Testing (SAT'23)* (Dagstuhl, Germany, 2023),

M. Mahajan and F. Slivovsky, Eds., vol. 271 of *Leibniz International Proceedings in Informatics (LIPIcs)*, Dagstuhl Publishing, pp. 6:1–6:20.

[10] Capelli, F. Knowledge compilation languages as proof systems. In *Proceedings of the 22nd International Conference on Theory and Applications of Satisfiability Testing (SAT'19)* (July 2019), M. Janota and I. Lynce, Eds., vol. 11628 of *Lecture Notes in Computer Science*, Springer Verlag, pp. 90–99.

[11] Capelli, F., Choi, Y., Mengel, S., Muñoz, M., and den Broeck, G. V. A canonical generalization of OBDD, 2026.

[12] Capelli, F., Pia, A. D., and Gregorio, S. D. A knowledge compilation take on binary polynomial optimization, 2024.

[13] Chakraborty, S., Meel, K. S., and Vardi, M. Y. Improving approximate counting for probabilistic inference: From linear to logarithmic SAT solver calls. In *Proceedings of 25th International Joint Conference on Artificial Intelligence (IJCAI'16)* (July 2016), S. Kambhampati, Ed., The AAAI Press, pp. 3569–3576.

[14] Chede, S., Chew, L., and Shukla, A. Circuits, proofs and propositional model counting. In *Proceedings of the 44th IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS'24)* (2024), S. Barman and S. Lasota, Eds., vol. 323 of *Leibniz International Proceedings in Informatics (LIPIcs)*, Schloss Dagstuhl – Leibniz-Zentrum für Informatik, pp. 18:1–18:23.

[15] Dahllöf, V., Jonsson, P., and Wahlström, M. Counting models for 2SAT and 3SAT formulae. *Theoretical Computer Science 332*, 1 (2005), 265–291.

[16] Darwiche, A. New advances in compiling CNF to decomposable negation normal form. In *Proceedings of the 16th European Conference on Artificial Intelligence (ECAI'04)* (2004), R. López De Mántaras and L. Saitta, Eds., IOS Press, pp. 318–322.

[17] Darwiche, A., and Marquis, P. A knowledge compilation map. *Journal of Artificial Intelligence Research 17* (2002), 229–264.

[18] De Raedt, L., Kimmig, A., and Toivonen, H. Problog: A probabilistic prolog and its application in link discovery. In *Proceedings of the 20th international joint conference on artificial intelligence (IJCAI'07)* (2007), IJCAI, pp. 2462–2467.

[19] Dudek, J. M., Phan, V. H. N., and Vardi, M. Y. ADDMC: Weighted model counting with algebraic decision diagrams. In *Proceedings of the 24th AAAI Conference on Artificial Intelligence AAAI'20* (2020), V. Conitzer and F. Sha, Eds., pp. 1468–1476.

[20] Dueñas-Osorio, L., Meel, K. S., Paredes, R., and Vardi, M. Y. Counting-based reliability estimation for power-transmission grids. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence (AAAI'17)* (San Francisco, CA, USA, Feb. 2017), S. P. Singh and S. Markovitch, Eds., The AAAI Press, pp. 4488–4494.

[21] Dueñas-Osorio, L., Meel, K. S., Paredes, R., and Vardi, M. Y. Counting-based reliability estimation for power-transmission grids. In *Proceedings of the 31st AAAI Conference on Artificial Intelligence (AAAI'17)* (2017), S. P. Singh and S. Markovitch, Eds., The AAAI Press, pp. 4488–4494.

[22] Eiter, T., Hecher, M., and Kiesel, R. Treewidth-Aware Cycle Breaking for Algebraic Answer Set Counting. In *Proceedings of the 18th International Conference on Principles of Knowledge Representation and Reasoning (KR'21)* (11 2021), pp. 269–279.

[23] Fichte, J. K., Gaggl, S. A., and Rusovac, D. Rushing and strolling among answer sets – navigation made easy. In *Proceedings of the 36th AAAI Conference on Artificial Intelligence (AAAI'22)* (2022), pp. 5651–5659.

[24] Fichte, J. K., and Hecher, M. The model counting competitions 2021-2023, 2025.

[25] Fichte, J. K., Hecher, M., and Hamiti, F. The model counting competition 2020. *ACM Journal of Experimental Algorithmics 26* (Oct. 2021).

[26] Fichte, J. K., Hecher, M., and Roland, V. Proofs for Propositional Model Counting. In *Proceedings of the 25th International Conference on Theory and Applications of Satisfiability Testing (SAT'22)* (Dagstuhl, Germany, 2022), K. S. Meel and O. Strichman, Eds., vol. 236 of *Leibniz International Proceedings in Informatics (LIPIcs)*, Dagstuhl Publishing, pp. 30:1–30:24.

[27] Fremont, D. J. *Algorithmic Improvisation.* PhD thesis, University of California, Berkeley, 2019.

[28] Hecher, M., Shaw, A., and Fichte, J. K. mccompetition.org. `https://mccompetition.org/`, 2026.

[29] Huang, J., and Darwiche, A. DPLL with a trace: from sat to knowledge compilation. In *Proceedings of the 19th International Joint Conference on Artificial Intelligence (IJCAI'05)* (San Francisco, CA, USA, 2005), L. P. Kaelbling and A. Saffiotti, Eds., IJCAI'05, Morgan Kaufmann Publishers Inc., pp. 156–162.

[30] Kiesel, R., and Eiter, T. Knowledge Compilation and More with SharpSAT-TD. In *Proceedings of the 20th International Conference on Principles of Knowledge Representation and Reasoning (KR'23)* (8 2023), pp. 406–416.

[31] Korhonen, T., and Järvisalo, M. Integrating Tree Decompositions into Decision Heuristics of Propositional Model Counters. In *Proceedings of the 27th International Conference on Principles and Practice of Constraint Programming (CP'21)* (2021), L. D. Michel, Ed., vol. 210 of *Leibniz International Proceedings in Informatics (LIPIcs)*, Dagstuhl Publishing, pp. 8:1–8:11.

[32] Korhonen, T., and Järvisalo, M. SharpSAT-TD in model counting competitions 2021-2023, 2023.

[33] Lagniez, J., Lonca, E., and Marquis, P. Improving model counting by leveraging definability. In *Proceedings of 25th International Joint Conference on Artificial Intelligence (IJCAI'16)* (July 2016), S. Kambhampati, Ed., The AAAI Press, pp. 751–757.

[34] Lagniez, J.-M., and Marquis, P. An improved decision-DDNF compiler. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence (IJCAI'17)* (2017), C. Sierra, Ed., The AAAI Press, pp. 667–673.

[35] Latour, A. L. D., Babaki, B., Fokkinga, D., Anastacio, M., Hoos, H. H., and Nijssen, S. Exact stochastic constraint optimisation with applications in network analysis. *Artificial Intelligence 304* (2022), 103650.

[36] Lee, J., and Yang, Z. LPMLN weak constraints, and P-log. In *Proceedings of the 31st AAAI Conference on Artificial Intelligence (AAAI'17)* (2017), AAAI Press, pp. 1170–1177.

[37] Manhaeve, R., Dumančić, S., Kimmig, A., Demeester, T., and De Raedt, L. Neural probabilistic logic programming in deepproblog. *Artificial Intelligence 298* (2021), 103504.

[38] Mei, J., Bonsangue, M., and Laarman, A. Simulating quantum circuits by model counting. In *Proceedings of the 36th International Conference Computer Aided Verification (CAV'24)* (2024), A. Gurfinkel and V. Ganesh, Eds., Springer Nature Switzerland, pp. 555–578.

[39] Muise, Christian J .and McIlraith, S. A., Beck, J. C., and Hsu, E. I. Dsharp: Fast d-DNNF compilation with sharpSAT. In *Proceedings of the 25th Canadian Conference on Artificial Intelligence (AI'17)* (2012), L. Kosseim and D. Inkpen, Eds., vol. 7310 of *Lecture Notes in Computer Science*, Springer Verlag, pp. 356–361.

[40] Oztok, U., and Darwiche, A. A top-down compiler for sentential decision diagrams. In *Proceedings of the 24th International Joint Conference on Artificial Intelligence (IJCAI'15)* (2015), Q. Yang and M. Wooldridge, Eds., The AAAI Press, pp. 3141–3148.

[41] Ramprasath, S., Charan, M. G. K. S., and Vasudevan, V. Exact computation of error in approximate circuits using SAT and message-passing algorithms, 2024.

[42] Roth, D. On the hardness of approximate reasoning. *Artif. Intell. 82*, 1-2 (1996), 273–302.

[43] Sharma, S., Roy, S., Soos, M., and Meel, K. S. GANAK: A scalable probabilistic exact model counter. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence (IJCAI'19)* (2019), S. Kraus, Ed., IJCAI.org, pp. 1169–1176.

[44] Shi, W., Shih, A., Darwiche, A., and Choi, A. On tractable representations of binary neural networks. In *Proceedings of the 17th International Conference on Principles of Knowledge Representation and Reasoning (KR'20)* (2020), D. Calvanese, E. Erdem, and M. Thielscher, Eds., pp. 882–892.

[45] Skryagin, A., Ochs, D., Dhami, D. S., and Kersting, K. Scalable neural-probabilistic answer set programming. *Journal of Artificial Intelligence Research 78* (2023), 579–617.

[46] Sundermann, C., Thüm, T., and Schaefer, I. Evaluating #SAT solvers on industrial feature models. In *Proceedings of the 14th International Working Conference on Variability Modelling of Software-Intensive Systems (VAMOS '20)* (2020), Assoc. Comput. Mach., New York.

[47] Teuber, S., and Weigl, A. Quantifying software reliability via model-counting. In *Proceedings of the 18th International Conference on Quantitative Evaluation of Systems (QEST'21)* (2021), A. Abate and A. Marin, Eds., Springer Verlag, pp. 59–79.

[48] Toda, S. PP is as hard as the polynomial-time hierarchy. *SIAM J. Comput. 20*, 5 (Oct. 1991), 865–877.

[49] Valiant, L. G. The complexity of computing the permanent. *Theoretical Computer Science 8*, 2 (1979), 189–201.

[50] Yang, Z., Ishay, A., and Lee, J. NeurASP: embracing neural networks into answer set programming. In *Proceedings of the 29th International Joint Conference on Artificial Intelligence (IJCAI'20)* (2021).

[51] Zhai, E., Chen, A., Piskac, R., Balakrishnan, M., Tian, B., Song, B., and Zhang, H. Check before you change: Preventing correlated failures in service updates. In *Proceedings of the 17th USENIX Symposium on Networked Systems Design and Implementation (NSDI'20)* (2020), R. Bhagwan and G. Porter, Eds., USENIX Association, pp. 575–589.