Research Statement

Arijit Shaw

Ph.D. Student, IAI, TCG CREST

1 Coursework

During my first year at IAI, I have attended few courses, both at TCG and at CMI. My marks are shown in table 1a and table 1b.

(a) Courses at First Semester

Course	Grade
Discrete Mathematics	79/100
Cryptology-I	80/100
Automata & Formal Lang.	87/100
AIML	82/100

(b) Courses at Second Semester

Course	Grade
Advanced Cryptology	57/100
Quantum Information & Cryptology	86/100
Design & Analysis Of Algorithm	91/100
Complexity Theory (CMI)	8/10
Research Methodology	82/100

2 Overview

I'm working on **building algorithms and software for counting over SMT constraints** with Dr. Kuldeep S. Meel, assistant professor at National University of Singapore. In simple terms, the problem I'm currently working on is constrained counting. Here, the task is to count the cardinality of the set of solutions of input constraints.

Complexity wise, the problem of constrained counting is #P-complete, even if we restrict the constraints to boolean constraints. The success of SAT in early 2000s inspired the quest to lift satisfiability techniques to more expressive theories to handle more expressive constraints such as linear real arithmetic, bitvectors, strings; such constraints allow precise modeling of modern hardware and software. These efforts have yielded in a ecosystem with availability of state of the art Satisfiability Modulo Theory (SMT) techniques that serve as crucial engines in modern formal methods and artificial intelligence.

A demand for a constrained counting tool over different theories has already been raised by different fields of Computer Science, like software verification [23] , cryptography [3] and computational biology [19]. Still there is no such tool that tackles the problem. Therefore, I plan to work on building such a counting engine during my PhD. We believe that we can solve problems from many other domains once we have such a counting tool is available.

3 Literature Review

As there is no similar work done in the exact problem I'm working on, I read papers from related fields. We can divide them into the following categories.

Model Counting is the problem of counting the number of solutions of a given boolean formula. There are a good number of solvers, some of them count top-down e.g., SharpSAT [24] and Ganak [20] , while others use a decision diagram based bottom-up approach, e.g., ADDMC [8] and DPMC [9]. Some of the counters include tree-decomposition based approach too [14, 11].

Approximate Model Counting gets relevant, when we are allowed to model count with some error bound. ApproxMC [5] and its following versions [6, 22] has a good line of research in this. These approximate counters has seen some interesting application in cryptography [3], reliability estimation [10], synthesis [13] and verification of neural nets [2].

SMT Solving is the decision problem for SMT formulas. The relevant theory is well established in the book[15], while the relevant tool papers of Boolector [4] and STP [12] discuss more on implementation.

Counting / integrating over theories. Lattice point enumeration tool, LattE [7] can count over integer arithmatic. Also there are tools for weighted model integration [18] that solves the problem of integraion for real arithmatic. Counting over string constraints became possible with the tool ABC [1].

Possible Application of SMT Counting that we are designing can range in everywhere where a model counter is being used, but the problem comes from an SMT domain. Automating CCA in cryptography [3], summarizing transmission trees in computational biology [19], quantitative software verification [23] – are to name few of them.

SAT solvers being the backbone of most of the counters, I am often mesmerized by the power and mystery they posses. As side projects, I look into works [16, 17, 21] that try to explore the power of solvers using AI.

4 Research Done

As of now, I have worked on the algorithmic foundation of such a counting engine and built a tool that can count over boolean + bitvector theory. And that tool is giving better results in comparison to existing techniques. Our system explores the power of advancements in knowledge compilation and SMT tenchniques. We plan to submit our work in upcoming conference – Constraint Programming (CP).

5 Research Plans

In the coming days, I plan to extend the tool for other theories like linear real arithmetic, integer arithmetic and strings. I'd also like to work on solving hard problems from cryptography and software verification that were not scalable without such tools.

The initial progress highlights the opportunities and I am excited to pursue this as a long-term research work in my PhD.

References

[1] Abdulbaki Aydin, Lucas Bang, and Tevfik Bultan. Automata-based model counting for string constraints. In *International Conference on Computer Aided Verification*, pages 255–272. Springer, 2015.

[2] Teodora Baluta, Shiqi Shen, Shweta Shinde, Kuldeep S Meel, and Prateek Saxena. Quantitative verification of neural networks and its security applications. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, pages 1249–1264, 2019.

[3] Gabrielle Beck, Maximilian Zinkus, and Matthew Green. Automating the development of chosen ciphertext attacks. In *29th {USENIX} Security Symposium ({USENIX} Security 20)*, pages 1821–1837, 2020.

[4] Robert Brummayer and Armin Biere. Boolector: An efficient smt solver for bit-vectors and arrays. In *International Conference on Tools and Algorithms for the Construction and Analysis of Systems*, pages 174–177. Springer, 2009.

[5] Supratik Chakraborty, Kuldeep S Meel, and Moshe Y Vardi. A scalable approximate model counter. In *International Conference on Principles and Practice of Constraint Programming*, pages 200–216. Springer, 2013.

[6] Supratik Chakraborty, Kuldeep S Meel, and Moshe Y Vardi. Algorithmic improvements in approximate counting for probabilistic inference: from linear to logarithmic sat calls. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, pages 3569–3576, 2016.

[7] Jesús A De Loera, Raymond Hemmecke, Jeremiah Tauzer, and Ruriko Yoshida. Effective lattice point counting in rational convex polytopes. *Journal of symbolic computation*, 38(4):1273–1302, 2004.

[8] Jeffrey M. Dudek, Vu H.N. Phan, and Moshe Y. Vardi. ADDMC: Weighted model counting with algebraic decision diagrams. *AAAI 2020 - 34th AAAI Conference on Artificial Intelligence*, pages 1468–1476, 2020.

[9] Jeffrey M Dudek, Vu HN Phan, and Moshe Y Vardi. Dpmc: Weighted model counting by dynamic programming on project-join trees. In *International Conference on Principles and Practice of Constraint Programming*, pages 211–230. Springer, 2020.

[10] Leonardo Duenas-Osorio, Kuldeep Meel, Roger Paredes, and Moshe Vardi. Counting-based reliability estimation for power-transmission grids. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 31, 2017.

[11] Johannes K Fichte, Markus Hecher, Stefan Woltran, and Markus Zisser. Weighted model counting on the gpu by exploiting small treewidth. In *26th Annual European Symposium on Algorithms (ESA 2018)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2018.

[12] Vijay Ganesh and David L Dill. A decision procedure for bit-vectors and arrays. In *International conference on computer aided verification*, pages 519–531. Springer, 2007.

[13] Priyanka Golia, Subhajit Roy, and Kuldeep S Meel. Manthan: a data-driven approach for boolean function synthesis. In *International Conference on Computer Aided Verification*, pages 611–633. Springer, 2020.

[14] Tuukka Korhonen and Matti Järvisalo. Sharpsat-td: Improving sharpsat by exploiting tree decompositions. 2021.

[15] Daniel Kroening and Ofer Strichman. *Decision procedures*. Springer, 2016.

[16] Vitaly Kurin, Saad Godil, Shimon Whiteson, and Bryan Catanzaro. Can q-learning with graph networks learn a generalizable branching heuristic for a sat solver? *Advances in Neural Information Processing Systems*, 33:9608–9621, 2020.

[17] Gil Lederman, Markus N Rabe, Edward A Lee, and Sanjit A Seshia. Learning heuristics for quantified boolean formulas through deep reinforcement learning. *arXiv preprint arXiv:1807.08058*, 2018.

[18] Paolo Morettin, Andrea Passerini, and Roberto Sebastiani. Advanced smt techniques for weighted model integration. *Artificial Intelligence*, 275:1–27, 2019.

[19] Palash Sashittal and Mohammed El-Kebir. Sampling and summarizing transmission trees with multi-strain infections. *Bioinformatics*, 36(Supplement_1):i362–i370, 2020.

[20] Shubham Sharma, Subhajit Roy, Mate Soos, and Kuldeep S Meel. Ganak: A scalable probabilistic exact model counter. In *IJCAI*, volume 19, pages 1169–1176, 2019.

[21] Mate Soos, Raghav Kulkarni, and Kuldeep S Meel. Crystalball: Gazing in the black box of sat solving. In *International Conference on Theory and Applications of Satisfiability Testing*, pages 371–387. Springer, 2019.

[22] Mate Soos and Kuldeep S Meel. Bird: engineering an efficient cnf-xor sat solver and its applications to approximate model counting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 1592–1599, 2019.

[23] Samuel Teuber and Alexander Weigl. Quantifying software reliability via model-counting. In Alessandro Abate and Andrea Marin, editors, *Quantitative Evaluation of Systems*, pages 59–79, Cham, 2021. Springer International Publishing.

[24] Marc Thurley. sharpsat–counting models with advanced component caching and implicit bcp. In *International Conference on Theory and Applications of Satisfiability Testing*, pages 424–429. Springer, 2006.