Mallob: Scalable SAT Solving On Demand With Decentralized Job Scheduling

Peter Sanders and Dominik Schreiber
1 Karlsruhe Institute of Technology, Germany

DOI: 10.21105/joss.04591

Summary

Propositional satisfiability (SAT) is the problem of finding a variable assignment for a given propositional formula (i.e., a composition of Boolean variables using logical operators NOT, AND, OR) such that the formula evaluates to true, or reporting that no such assignment exists. The platform Mallob (Malleable Load Balancer, or Massively Parallel Logic Backend) enables processing SAT jobs in a (massively) parallel and distributed system on demand. Mallob’s flexible, fair, and decentralized approach to online job scheduling results in scheduling latencies in the range of milliseconds, near-optimal system utilization, and high resource efficiency.

Statement of need

Despite SAT being a notoriously difficult problem (S. A. Cook, 1971), practically efficient SAT solving approaches have empowered a wide range of real-world applications for SAT such as software verification (Kleine-Büning et al., 2019), circuit design (Goldberg et al., 2001), cryptography (Massacci & Marraro, 2000), automated planning (Schreiber, 2021a), and theorem proving (Heule et al., 2016). With respect to modern computing paradigms such as cloud computing and high-performance computing (HPC), the limited scalability of established parallel and distributed SAT solvers has become a pressing issue (Hamadi & Wintersteiger, 2012). Moreover, since processing times of SAT jobs are unknown in advance, conventional HPC scheduling approaches applied to such jobs can lead to prohibitively large scheduling latencies and to suboptimal utilization of computational resources. Instead, we suggest making use of so-called malleable scheduling. A parallel computing task is called malleable if the amount of computational resources allotted (i.e., the number of cores or machines) can be adjusted during its execution (Feitelson, 1997). Malleability is a powerful paradigm in the field of job scheduling and load balancing as it allows scheduling incoming jobs rapidly and continuously rebalancing the present tasks according to their priority and other properties.

We believe that a cloud service that combines a scalable SAT solving engine with malleable job scheduling can significantly improve many SAT-related academic and industrial workflows in terms of productivity and (resource-)efficiency. Moreover, our decentralized scheduling approach is applicable to further applications beyond SAT where processing times are unknown in advance and a modest amount of data is transferred between the workers.

System overview

Mallob is a C++ application designed for parallel and distributed systems between 16 and 10000 cores; it allows resolving propositional problems in a (massively) parallel manner. Our SAT solving engine scales up to two thousand cores (Schreiber & Sanders, 2021) using a portfolio of established sequential SAT solvers (Audemard & Simon, 2009; Biere, 2017; Biere et al., 2020) and a careful exchange of information among them. Our submissions of Mallob to...
the International SAT Competitions (Schreiber, 2020, 2021b) won multiple prizes (Balyo et al., 2021; Froleyks et al., 2021). Following this success, Mallob has been referred to as “by a wide margin, the most powerful SAT solver on the planet” (B. Cook, 2022). Each distributed SAT solving task is malleable, allowing users to submit formulae to Mallob at will, with scheduling latencies in the range of milliseconds (Sanders & Schreiber, 2022b). Computational resources are allotted proportionally to each job’s priority and also respecting each job’s (maximum) demand for resources. Our scheduling approach is fully decentralized and uses a small part of each worker’s CPU time (< 5%) to perform scheduling negotiations. As such, Mallob achieves optimal system utilization, i.e., either all processes are utilized or all resource demands are fully met. We achieve this feat by arranging each active job as a binary tree of workers and growing or shrinking each job tree dynamically based on the current system state. For an in-depth discussion of these techniques, we refer to Schreiber & Sanders (2021), Sanders & Schreiber (2022a), and Sanders & Schreiber (2022b).

The further development of Mallob is an ongoing effort. As such, we are in the process of integrating engines for NP-hard applications beyond SAT, such as k-Means clustering or hierarchical planning, into Mallob.

Acknowledgements

We wish to thank the numerous people whose code we make thankful use of, including Balyo et al. (2015), Biere et al. (2020), Audemard & Simon (2009), Eén & Sörensson (2003), Lohmann (2022), Goetghebuer-Planchon (2022), Ankerl (2022), and Dusíková (2022). The authors gratefully acknowledge the Gauss Centre for Supercomputing e.V. (www.gauss-centre.eu) for funding this project by providing computing time on the GCS Supercomputer SuperMUC-NG at Leibniz Supercomputing Centre (www.lrz.de). Moreover, some of this work was performed on the HoreKa supercomputer funded by the Ministry of Science, Research and the Arts Baden-Württemberg and by the Federal Ministry of Education and Research. This project has received funding from the European Research Council (ERC) under the European Union’s Horizon 2020 research and innovation programme (grant agreement No. 882500).


