Using reinforcement learning for constraint solving

Anonymous author
Anonymous affiliation

Abstract

It is well known from the SAT/CP community that different strategies for selecting variables during search (i.e., the order in which the variables are assigned) exhibit different performance depending on the instance or class of problem they solve. Numerous such strategies exist in the literature, including static, dynamic and adaptive ones. Choosing the best strategy requires domain expertise, which deprives many users of the best choice for their problem. In this doctoral thesis, we aim to investigate and formally capture properties of existing variable-selection strategies to better understand their relationship and interaction with other important parameters in CSP solving, such as randomness, constraint propagation, or no-good recording. Learning frameworks are also utilized to identify the most suitable methods for the task, in particular various variants of the multi-armed bandit model. The ultimate goal is to propose a new learning framework that can guide search by efficiently combining branching strategies and propagation on-line.

2012 ACM Subject Classification Theory of computation → Constraint and logic programming

Keywords and phrases Constraint Programming; Branching Heuristics; Multi-Armed Bandits; Online Learning; Adaptive Search

Digital Object Identifier 10.4230/LIPIcs.CVIT.2016.23

1 Introduction

Constraint programming (CP) solves combinatorial problems by alternating variable selection and propagation. At each decision step, the solver chooses a variable, assigns a value, and propagates the consequences of that decision. The performance of the solver depends strongly on the branching strategy, but also on the amount of propagation, restart policies, and the management of learned information through nogoods.

This thesis is concerned with online adaptation of solver decisions. The central question is not whether a heuristic is good in absolute terms, but when, why, and under which search conditions it is useful.

A key difficulty is that existing work often studies branching, propagation, restarts, or nogood recording separately, even though each of these aspects is already difficult to analyze in isolation, whereas their combined effect determines the behaviour of the solver during search. As a result, a strategy that looks strong in isolation may become less effective once stronger propagation, restart policies, or learned nogoods change the search dynamics. This gap motivates a study of branching strategies as adaptive objects rather than fixed components. The abstract reflects this goal: the focus is on variable-selection strategies, their interaction with other solver parameters, and the use of multi-armed bandits to guide search online.

2 Variable-Selection Strategies

Variable-selection strategies are a core component of CP search. Static strategies fix the order of variables before search starts. Dynamic strategies update their choice during the search using information collected from the current state. Adaptive strategies go further and modify the strategy itself according to the observed behaviour of the solver.

42nd Conference on Very Important Topics (CVIT 2016).
Editors: John Q. Open and Joan R. Access; Article No. 23; pp. 23:1–23:6
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Classical examples include minimum-domain and degree-based rules, which embody the first-fail principle. More informative heuristics such as `dom/wdeg` [5] exploit failure history by increasing the weight of constraints involved in domain wipeouts. Other adaptive approaches use impact [13], fails [9, 5, 16] or activity [11] to diversify the search and to avoid being trapped in unproductive regions of the tree.

The main difficulty is that the quality of a branching heuristic is highly instance-dependent. A heuristic that performs well on one class of problems may be inferior on another, and the relative ranking of heuristics may even change during the search. Although many heuristics proposed in the literature obtain excellent performance on particular classes of instances, no single strategy consistently outperforms all others across the full diversity of CSP problems. This dependence is exactly what motivates a learning-based framework: the solver should not commit to one strategy globally, but should adapt it during search.

3 Interaction with Propagation, Restarts, and Nogood Recording

3.1 Propagation and consistency levels

The effect of branching cannot be separated from the rest of the solver. Propagation changes the size and shape of the remaining search space, and therefore changes the apparent quality of a variable-selection strategy. Stronger propagation can reduce the number of nodes, but it can also increase the cost of each node. This creates a trade-off between pruning power and inference cost.

The most common form of propagation in CP is Generalized Arc Consistency (GAC). Intuitively, a CSP is GAC if every value of every variable has a support in each constraint involving this variable. During propagation, values that no longer have any support are removed from the domains. GAC is widely used because it provides an effective compromise between pruning and computational overhead.

Stronger consistencies have also been proposed in order to reduce the search tree more aggressively. Singleton Arc Consistency (SAC) is a classical example. A value is singleton arc consistent if temporarily assigning this value and enforcing GAC does not lead to a contradiction. SAC is therefore able to detect inconsistencies that remain invisible under plain GAC. However, this stronger filtering comes at a much higher computational cost because propagation must be repeated for many tentative assignments.

These consistency levels strongly influence the behaviour of branching heuristics. Some heuristics may benefit from strong propagation because failures are detected earlier, while others may become less informative once domains are heavily filtered. Moreover, their effectiveness may vary depending on the depth in the search tree [6]. Understanding these interactions is one of the motivations of this thesis, because the right branching choice depends not only on the instance but also on the propagation regime.

Many approaches have attempted to adapt consistency levels dynamically during search, as in [4, 3]. Nevertheless, it remains unclear whether the additional pruning provided by stronger consistencies systematically compensates for their computational cost across different classes of CSP instances.

3.2 Restarts and heavy-tail behaviour

Restart policies also play an important role in modern CP and SAT solvers. Search procedures often exhibit heavy-tail behaviour, meaning that two runs of the same solver on the same instance may require dramatically different runtimes depending on early branching decisions.

Restarts mitigate this phenomenon by periodically interrupting the search and restarting [8] exploration from the root of the tree. The objective is to avoid spending excessive time in unproductive parts of the search space while preserving useful information gathered during previous attempts.

Restart strategies are now standard in SAT and CP solving because they improve robustness and interact well with learning mechanisms. However, they also modify the dynamics of search. A branching heuristic that performs poorly during one restart phase may become effective after the solver has accumulated additional information. This makes restart-based search particularly relevant when studying adaptive branching, since the same heuristic must be evaluated under changing conditions.

3.3 Nogood recording and learning

Nogood recording adds another layer of interaction. A nogood is a combination of assignments known to be inconsistent. Recording nogoods prevents the solver from revisiting previously explored failing configurations.

One common approach consists in recording nogoods across restarts [10]. Learned nogoods are preserved between restart episodes and reused during subsequent explorations of the search tree. As the search progresses, the solver accumulates information about conflicts and progressively refines the explored search space.

Another important framework is Lazy Clause Generation (LCG), which combines finite-domain propagation with SAT-style clause learning [7]. In LCG, propagators must explain their inferences through clauses or nogoods. These explanations allow the solver to derive learned clauses and perform conflict-directed backtracking.

LCG provides a powerful integration between propagation and learning, but it also increases implementation complexity. Strong propagators and strong consistencies are often difficult to explain efficiently. As a consequence, the interaction between propagation strength, branching heuristics, and nogood recording becomes highly non-trivial. In practice, this means that one cannot evaluate branching strategies independently of the propagation and learning machinery that surrounds them.

4 Online Adaptation with Multi-Armed Bandits

Multi-Armed Bandit (MAB) algorithms provide a natural framework for online decision making. At each step, the solver selects one arm, observes a reward, and updates its beliefs about which decisions are likely to be useful. The objective is to balance exploration and exploitation.

This framework is well suited to CP [15, 12, 4] because the best branching strategy, like the most appropriate propagation level, is generally not known in advance. Moreover, the usefulness of a heuristic is rarely stationary during search. A strategy may be effective early in the exploration, then become less relevant after propagation has reduced the domains, after restarts have modified the search trajectory, or after nogoods have accumulated. MAB methods are therefore attractive because they can adapt decisions online without requiring an explicit model of the instance.

Several families of bandit algorithms are relevant in this context. UCB-style methods like UCB1 [1] are attractive when rewards are noisy but relatively stable. Thompson Sampling [14] offers a Bayesian alternative, whereas EXP3 [2] is more suitable when rewards evolve in a highly non-stationary or adversarial way. In CP, these algorithms may be applied at different granularities. A bandit policy may select a heuristic at each restart, meaning that the same heuristic is used during an entire restart episode, or it may be invoked dynamically at each node during search. The appropriate level of adaptation depends on the stability of the search dynamics and on the computational overhead induced by the learning process.

4.1 Definition of the arms

The definition of the arms is a central design decision. In this thesis, the most natural arms correspond to variable-selection strategies, such as `dom/wdeg`, impact-based search, activity-based search, or randomized variants. A more expressive formulation may combine branching with value selection and propagation settings, so that an arm represents a complete solver configuration.

A coarse-grained arm set simplifies learning but may miss useful structure. A fine-grained arm set increases flexibility but can make learning unstable. The thesis will study this trade-off explicitly.

4.2 Reward design

The reward must reflect the usefulness of a decision during search. Candidate rewards include the reduction in subtree size, the amount of pruning caused by propagation, the frequency of failures, or normalized measures of search effort. The important point is that the reward should be aligned with solving progress, not merely with local activity.

However, defining a good reward is itself a difficult and largely open problem. In practice, it is often unclear what constitutes a good decision during search. A branching choice that appears beneficial locally may lead to poor global behaviour later in the exploration, whereas an apparently expensive propagation step may drastically reduce the remaining search tree. As a consequence, reward functions generally capture only partial approximations of the true usefulness of a decision.

The definition of the reward also depends on the granularity of adaptation. If rewards are measured over an entire restart episode, they may better reflect the global behaviour of a heuristic during search. Conversely, more local rewards may react faster to changes in the explored search space. Similarly, the bandit policy itself may be applied at different scales, for example at each node or only at restart boundaries. Both options are relevant, and both will be considered in the thesis.

5 Research Questions

The thesis is structured around four questions:

1. Which properties best explain the behaviour of search strategies across instances and problem classes?
2. How do branching strategies, consistency levels, restarts, and nogood recording interact during search?
3. Which bandit formulations are the most appropriate for adapting search strategies online in CP?
4. Which reward definitions and adaptation granularities best capture the evolution of search behaviour during solving?

6 Expected Contributions

The expected contributions are the following, with branching strategies as the main object of analysis:

- a formal characterization of search strategies and their role in CP solving;
- an analysis of the interactions between branching heuristics, consistency levels, restarts, and nogood recording;
- a multi-armed bandit framework for the online selection of branching strategies;
- an extension of this framework toward the joint adaptation of branching and propagation mechanisms;
- an empirical study on benchmark families representative of different search behaviours and propagation regimes.

7 Conclusion

The objective of this doctoral work is to move from hand-tuned search policies to an online learning framework capable of adapting solver decisions during search. The long-term objective is to understand and combine branching, propagation, restarts, and nogood recording within a unified adaptive framework.

A first focus of the thesis is the online selection of branching strategies using multi-armed bandit methods. Branching heuristics provide a natural starting point because they are central to CP performance, highly instance-dependent, and strongly influenced by propagation and learning mechanisms.

Propagation, restarts, and nogood recording are therefore considered both as interacting components of the solver and as sources of information for online adaptation.

References

1 Peter Auer, Nicolò Cesa-Bianchi, and Paul Fischer. Finite-time analysis of the multiarmed bandit problem. *Mach. Learn.*, 47(2-3):235–256, 2002. `doi:10.1023/A:1013689704352`.

2 Peter Auer, Nicolò Cesa-Bianchi, Yoav Freund, and Robert E. Schapire. The non-stochastic multiarmed bandit problem. *SIAM J. Comput.*, 32(1):48–77, 2002. `doi:10.1137/S0097539701398375`.

3 Amine Balafrej, Christian Bessiere, El Houssine Bouyakhf, and Gilles Trombettoni. Adaptive singleton-based consistencies. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 28, 2014.

4 Amine Balafrej, Christian Bessiere, and Anastasia Paparrizou. Multi-armed bandits for adaptive constraint propagation. In Qiang Yang and Michael J. Wooldridge, editors, *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence, IJCAI 2015, Buenos Aires, Argentina, July 25-31, 2015*, pages 290–296. AAAI Press, 2015. URL: `http://ijcai.org/Abstract/15/047`.

5 Frédéric Boussemart, Fred Hemery, Christophe Lecoutre, and Lakhdar Sais. Boosting systematic search by weighting constraints. In Ramón López de Mántaras and Lorenza Saitta, editors, *Proceedings of the 16th Eureopean Conference on Artificial Intelligence, ECAI'2004, including Prestigious Applicants of Intelligent Systems, PAIS 2004, Valencia, Spain, August 22-27, 2004*, pages 146–150. IOS Press, 2004.

6 Romuald Debruyne. Etude des consistances locales pour les problemes de satisfaction de contraintes de grande taille. *These de l'Université de Montpellier II*, 1998.

7 Thibaut Feydy and Peter J Stuckey. Lazy clause generation reengineered. In *International conference on principles and practice of constraint programming*, pages 352–366. Springer, 2009.

8 Carla P Gomes, Bart Selman, Henry Kautz, et al. Boosting combinatorial search through randomization. *AAAI/IAAI*, 98(1998):431–437, 1998.

9 Djamal Habet and Cyril Terrioux. Conflict history based search for constraint satisfaction problem. In Chih-Cheng Hung and George A. Papadopoulos, editors, *Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing, SAC 2019, Limassol, Cyprus, April 8-12, 2019*, pages 1117–1122. ACM, 2019. doi:10.1145/3297280.3297389.

10 Christophe Lecoutre, Lakhdar Sais, Sébastien Tabary, Vincent Vidal, et al. Nogood recording from restarts. In *IJCAI*, volume 7, pages 131–136, 2007.

11 Hongbo Li, Minghao Yin, and Zhanshan Li. Failure based variable ordering heuristics for solving csps (short paper). In Laurent D. Michel, editor, *27th International Conference on Principles and Practice of Constraint Programming, CP 2021, Montpellier, France (Virtual Conference), October 25-29, 2021*, LIPIcs, pages 9:1–9:10. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021. URL: https://doi.org/10.4230/LIPIcs.CP.2021.9, doi:10.4230/LIPICS.CP.2021.9.

12 Anastasia Paparrizou and Hugues Wattez. Perturbing branching heuristics in constraint solving. In *International Conference on Principles and Practice of Constraint Programming*, pages 496–513. Springer, 2020.

13 Philippe Refalo. Impact-based search strategies for constraint programming. In Mark Wallace, editor, *Principles and Practice of Constraint Programming - CP 2004, 10th International Conference, CP 2004, Toronto, Canada, September 27 - October 1, 2004, Proceedings*, Lecture Notes in Computer Science, pages 557–571. Springer, 2004. doi:10.1007/978-3-540-30201-8_41.

14 William R Thompson. On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika*, 25(3/4):285–294, 1933.

15 Hugues Wattez, Frédéric Koriche, Christophe Lecoutre, Anastasia Paparrizou, and Sébastien Tabary. Learning variable ordering heuristics with multi-armed bandits and restarts. In *ECAI 2020-24th European Conference on Artificial Intelligence*, 2020.

16 Hugues Wattez, Christophe Lecoutre, Anastasia Paparrizou, and Sébastien Tabary. Refining constraint weighting. In *31st IEEE International Conference on Tools with Artificial Intelligence, ICTAI 2019, Portland, OR, USA, November 4-6, 2019*, pages 71–77. IEEE, 2019. doi:10.1109/ICTAI.2019.00019.