Searching with Distributional Heuristics (Student Abstract)

Stephen Wissow*
University of New Hampshire
sjw@cs.unh.edu

Abstract

Distributional heuristic search uses distributions rather than point values for its heuristic estimates of the cost to goal from any given state. Distributional heuristics are desirable as they provide search algorithms not only with a way to evaluate nodes, but also with a basis for rational decision making tailored to specific search settings. Bounded suboptimal, anytime, and contract searches have differing but related objectives that each lend themselves to probabilistic reasoning supported by distributional heuristics. In many applications, speed of planning can be more important than solution quality. Whether due to certain domains' inherent difficulty, where anything but a satisficing approach is infeasible due to time or memory constraints, or due to the limited planning time available in real-time robotics and other time-sensitive planning settings, important open questions are how best to find solutions as quickly as possible and how to find the best solution possible while subject to an explicit limit on planning time. Successful algorithms must reason not only about solution cost, possibly in relation to a suboptimality bound, but also about the relative likelihood of finding a solution under one node vs. under another, of finding a solution of a particular cost (such as in relation to that of an incumbent solution), or about the expected amount of search effort to find a goal under a given node. This dissertation takes up these issues in four parts. I (1) examine different methods for generating distributional heuristics in bounded cost heuristic search and classical planning; (2) study the contract search setting, which involves online estimation of several unknown values; (3) consider the bounded suboptimal setting; and (4) address the anytime setting.

Sources of Distributional Heuristics

The problem of online generation of distributional heuristics has received little attention, and this only recently. Though I am more interested in online learning, the approach by Heller et al. (2022) of considering confidence when selecting nodes for expansion is promising. In the bounded cost heuristic search setting, Fickert, Gu, and Ruml (2021) synthesized a Gaussian distributional heuristic with a variance based on an estimate of single-step error learned online, and a mean based on the error-corrected value $\hat{f}$. However, in ongoing collaboration with Dr. Masataro Asai at the MIT-IBM Watson AI Lab in Cambridge, MA, USA, I have developed a new distributional heuristic for classical planning that works by explicitly randomizing the tie-breaking within the Fast Forward heuristic (Hoffmann and Nebel 2001, h_{FF}). Preliminary results confirm that, for any given state, our new heuristic, h_{FFrand}, may be sampled and those samples used to estimate the parameters of its distribution. As well as being data-driven, this approach is more principled than that of Fickert, Gu, and Ruml (2021), particularly with regard to estimating the heuristic's variance, since tie-breaking is inherent to how h_{FF} is computed. While this work is ongoing, we've also developed a new method (Wissow and Asai 2023) to take advantage of distributional heuristic information in the classical planning setting that is based on Monte Carlo Tree Search (MCTS) and which, our empirical evaluation shows, improves both upon the relevant previous work (Schulte and Keller 2014) and upon the traditional baseline of Greedy Best First Search (GBFS) in terms of coverage per expansion limit. It's still an important question whether the overhead of calculating distributional heuristics is worth it, and I am currently investigating this.

Contract Search

Contract search is a very important setting that has not received much attention, and I am developing a new algorithm that does not need to estimate as many parameters as the current state of the art. In contract search, the objective is to find the best (cheapest) solution possible subject to an absolute bound on planning time (or, for reproducibility, on node expansions). While contract search is thus a very common and intuitive problem setting, in practice it is instead anytime algorithms that have been used to solve contract search problems. However, anytime algorithms, such as the popular Anytime Repairing A* (Likhachev, Gordon, and Thrun 2003, ARA*), by design have zero knowledge of when they will be terminated by the user, and therefore cannot modulate their search behavior in response to a specific contract, let alone in an online manner as the contract's deadline approaches during planning. (I discuss the definition of the anytime setting in greater detail in its own section, below.)

Deadline-Aware Search (Dionne, Thayer, and Ruml 2011, DAS) is the current state of the art contract search algorithm, and proved that an algorithm can leverage knowledge of the

*Advisor: Wheeler Ruml

deadline to improve search performance. However, since it uses a best-first style priority queue of nodes, DAS must estimate two different parameters: average expansion delay and average single-step error in d (distance to goal in number of edges). Additionally, DAS frequently prunes all nodes as not leading to a goal within the remaining contract, and is repeatedly forced into its recovery mode in a pathological fashion in order to repopulate its open list with previously pruned nodes. To address these shortcomings, I am developing Deadline-Aware Beam Search (DABS), a new contract search algorithm based instead on beam search. Beam search can be understood as multiple, parallel hill-climbing, without a priority queue, obviating the need to estimate expansion delay. Whereas beam search traditionally uses a fixed beam width determined by the user at runtime, DABS adjusts its beam width online at each layer in relation to the time (or number of expansions) remaining in the contract. Nodes are selected by low d- rather than low f- or low h-value, as this has been shown to find better solutions faster in beam search (Lemons et al. 2022). Initial experiments show DABS' improvement over DAS, and this work is ongoing.

Bounded Suboptimal Search

In bounded suboptimal search, the objective is to find a solution whose cost is within a given constant factor $w > 1$ of the cost of the optimal solution g^*_{goal}. Weighted A* (Pohl 1970, wA*) is a popular bounded suboptimal search algorithm, and, when space and time constraints preclude optimal search, practitioners often attempt to decrease planning time by increasing w in wA*. Recent work in bounded suboptimal search fails to take sufficient advantage of distributional information when selecting which node to expand next. Fickert, Gu, and Ruml (2022)'s RR-d algorithm uses round-robin alternation between three different queues, and thus is predicated on the broad assumption that these queues have equivalent usefulness that is constant over the course of search. I will instead propose a bounded suboptimal search algorithm that uses distributional information to inform queue alternation.

Anytime Search

The reasoning central to anytime search is akin to that of bounded suboptimal search, and a distributional heuristic provides the kind of information needed by the rational anytime algorithm. However, while anytime algorithms, like ARA*, are common in practice, the objective of the anytime search setting begs refinement. In general, an anytime search algorithm should return a solution whenever the user chooses to terminate the run, and, while still running, should search to improve upon its incumbent solution. Thayer, Benton, and Helmert (2012) argued that the objective in the anytime setting is to minimize the time between solutions during a run. However, I contend that the magnitude of the decrease in solution cost per elapsed planning time is also an essential performance metric in the anytime setting. In practice, since the duration for which an anytime algorithm will be allowed to run is unknown, it is important to find the first solution as quickly as possible. For this reason, it makes sense for an initial speedy search (greedy on d) to be performed. Thus we can assume an incumbent solution is always known, and characterize the essential nature of the anytime problem setting as that of finding a solution of lower cost than the incumbent. Therefore, a rational anytime algorithm must reason not only about the probability that a node's subtree contains a solution of cost less than the incumbent, but also about the confidence in the heuristic value of that node in comparison to another.

Improved anytime and contract search algorithms promise new insight into online reasoning in heuristic search, with wide practical relevance for time-sensitive planning.

References

Dionne, A.; Thayer, J.; and Ruml, W. 2011. Deadline-Aware Search Using On-Line Measures of Behavior. In *Proceedings of the 4th Annual Symposium on Combinatorial Search*, 39–46.

Fickert, M.; Gu, T.; and Ruml, W. 2021. Bounded-cost Search Using Estimates of Uncertainty. In *Proceedings of the 30th International Joint Conference on Artificial Intelligence*, 1675–1681.

Fickert, M.; Gu, T.; and Ruml, W. 2022. New Results in Bounded-Suboptimal Search. In *Proceedings of the 36th AAAI Conference on Artificial Intelligence*, 10166–10173.

Heller, D.; Ferber, P.; Bitterwolf, J.; Hein, M.; and Hoffmann, J. 2022. Neural Network Heuristic Functions: Taking Confidence into Account. In *Proceedings of the 15th International Symposium on Combinatorial Search*, 223–228.

Hoffmann, J.; and Nebel, B. 2001. The FF Planning System: Fast Plan Generation Through Heuristic Search. *Journal of Artificial Intelligence Research*, 14: 253–302.

Lemons, S.; Linares López, C.; Holte, R. C.; and Ruml, W. 2022. Beam Search: Faster and Monotonic. In *Proceedings of the 32nd International Conference on Automated Planning and Scheduling*, 222–230.

Likhachev, M.; Gordon, G. J.; and Thrun, S. 2003. ARA*: Anytime A* with Provable Bounds on Sub-Optimality. In *Neural Information Processing Systems*, 767–774.

Pohl, I. 1970. Heuristic Search Viewed as Path Finding in a Graph. *Artif. Intell.*, 1(3): 193–204.

Schulte, T.; and Keller, T. 2014. Balancing Exploration and Exploitation in Classical Planning. In *Proceedings of the 7th Annual Symposium on Combinatorial Search*, 139–147.

Thayer, J. T.; Benton, J.; and Helmert, M. 2012. Better Parameter-Free Anytime Search by Minimizing Time Between Solutions. In *Proceedings of the 5th Annual Symposium on Combinatorial Search*, 120–128.

Wissow, S.; and Asai, M. 2023. Scale-Adaptive Balancing of Exploration and Exploitation in Classical Planning. Forthcoming.