Adaptive Width Best-First Search: Dynamic Frontier Expansion for GNN-based Heuristics (Extended Abstract)

Valerio Borelli[1], **Alfonso Emilio Gerevini**[1], **Enrico Scala**[1], **Ivan Serina**[1]

[1] University of Brescia

valerio.borelli@unibs.it, alfonso.gerevini@unibs.it, enrico.scala@unibs.it, ivan.serina@unibs.it

Abstract

In this paper, we introduce Adaptive Width Best-First Search (AWBFS), a search algorithm designed to better exploit batched heuristic evaluation while preserving the guidance of Greedy Best-First Search (GBFS) when using learning-based heuristics on GPU architectures. The method targets domains in which learned heuristics are effective, but standard GBFS can stagnate on heuristic plateaus, a situation that is common in numeric planning. AWBFS addresses this issue by dynamically adjusting the number of frontier nodes expanded at each step. Starting from standard GBFS behavior, the algorithm increases the expansion width when no improvement in heuristic value is observed. For the use with learning-based heuristics, AWBFS tracks GPU memory usage to ensure it never exceeds the input limit. Experimental results show that AWBFS yields better performance than standard GBFS when paired with GNN-based heuristics.

Introduction

Learning-based heuristics have been used in a variety of different approaches for classical planning, and more recently to other types of planning, with numeric planning being the most prominent (Chen and Thiébaux 2024; Borelli et al. 2026a). However, despite these advancements in heuristic design, these approaches predominantly continue to rely on standard Greedy Best-First Search (GBFS) (Bonet and Geffner 2001) as their primary search algorithm. Traditional search algorithms often treat the heuristic merely as a black box, failing to explore novel search strategies that could perform significantly better in contexts where heuristic evaluation heavily benefits from GPU acceleration. Because neural heuristics are highly optimized for batch processing on modern accelerators, sticking to a sequential node expansion strategy overlooks a critical opportunity to co-design the search algorithm with the hardware that executes it.

Methodology

The core innovation of Adaptive Width Best-First Search (AWBFS) is the dynamic adjustment of the search frontier's width based on the feedback from the heuristic functions. Unlike standard GBFS, which expands a single state per iteration, AWBFS maintains a dynamic parameter, called *width* (Felner, Kraus, and Korf 2003), which determines the number of nodes extracted from the Open List at each iteration.

This concept is inspired by K-Best-First Search (KBFS) (Felner, Kraus, and Korf 2003), which generalizes GBFS by expanding a fixed number of k nodes at each step. However, while KBFS uses a static *width*, AWBFS dynamically adjusts it, trying to find a dynamic balance based on the heuristic feedback.

Dynamic Width Adaptation The behaviour of the algorithm is governed by two phases, determined by the heuristic value of the expanded states.

GREEDY PHASE (DESCENT) At the start of the search, or whenever a new state s is found, such that $h(s) < besth$ (where $besth$ is the best heuristic value observed so far), the improvement in the heuristic value means that the current search trajectory is promising. Consequently, the algorithm prioritizes exploitation over exploration. In this phase, *width* is kept steady, mimicking the behaviour of standard GBFS when the *width* is at 1.

EXPLORATION PHASE (EXPANSION) If the heuristic value of the expanded nodes stops improving (i.e. $h(s) \geq besth$), the algorithm assumes that the search has encountered a plateau. To avoid getting stuck, the algorithm monotonically increases the *width* by 1. This allows the search to broaden its frontier, processing multiple candidate paths in parallel, in order to find an exit from the uninformative region.

Memory Constraints with Learning-based Heuristics The implementation of AWBFS is particularly synergistic with learning-based heuristics that leverage GPU acceleration. Since heuristic evaluation is performed via optimized batch processing, increasing the batch size incurs minimal impact on the time required per iteration.

AWBFS exploits this by processing a larger set of candidate states when needed, without significantly slowing down the search cycle. However, the batch size cannot be increased arbitrarily. It is necessary to strictly control the expansion to prevent excessive memory usage, ensuring that the search never exceeds the maximum available memory limit m. By balancing the computational throughput against the finite memory capacity, AWBFS ensures robust performance without risking resource exhaustion.

To correctly predict the amount of memory that the algo-

rithm will use before increasing the *width*, we need to calculate two things: the average memory cost per evaluation, and the estimated branching factor. The branching factor is the average number of applicable actions in a set of states (i.e. the average number of successors for each state).

Calculating the branching factor is not an easy task, as it is not a static property of the domain but varies significantly across different regions of the state space. To address this, AWBFS relies on an online estimation mechanism. Instead of using a pre-determined fixed value, the algorithm calculates the average branching factor based on the ratio of actual successors generated to the number of expanded nodes. This dynamic estimates allows the system to predict the memory usage of future evaluations, adapting the *width* constraint to the specific density of the task to be solved.

Then, before incrementing the *width*, the algorithm checks if increasing the batch size would exceed the memory limit m. The estimated memory usage is calculated as $mem \cdot (bf \cdot (width + 1))$, where mem represents the average memory cost per node, and bf is the estimated branching factor. This predictive check ensures that the expansion of the search frontier happens only if the predicted memory consumption remains within the available GPU memory.

Experimental Evaluation

We evaluate AWBFS on numeric planning problems, against a GPU-optimized baseline of standard GBFS, as defined in Borelli et al. (2026b). To ensure a fair comparison, this variant retains the exact structural properties and search behavior of the standard algorithm but is engineered to evaluate all successors of the expanded nodes in parallel. This adjustment allows the baseline to leverage the GPU's computational power effectively, ensuring that the comparison focuses on the algorithmic efficiency of the search strategy rather than implementation bottlenecks.

As numeric planning problems, we selected nine domains from the International Planning Competition (IPC) 2023 Numeric Track (Taitler et al. 2024): `Counters`, `Fo-Counters`, `Sailing`, `Fo-Sailing`, `Mprime`, `Expedition`, `Hydropower`, `Farmland`, and `Fo-Farmland`. We evaluate both algorithms with the GNN-based heuristic h^n(Borelli et al. 2026a), experiments are run on a single Intel Xeon Gold 6140M (2.30 GHz) core with a 5-minutes timeout for search. For the GPU, we use an NVIDIA v100 32GB GPU. We kept an 8GB GPU memory limit for AWBFS when running with the GPU, to mirror the standard 8GB memory limit typically enforced in CPU-based evaluations. The results show that AWBFS performs better than GBFS, outperforming the baseline in terms of coverage, speed, and solution quality.

Domain	$GBFS$			$AWBFS$		
	C	T	L	C	T	L
`Counters`	10	**13.89**	100.3	**11**	17.72	**94.2**
`Fo-Counters`	**8**	29.30	23.8	**8**	**1.30**	**17.0**
`Sailing`	**2**	**13.08**	**180.5**	**2**	14.59	182.5
`Fo-Sailing`	**8**	33.17	269.6	**8**	**29.97**	**269.5**
`Mprime`	**8**	**1.42**	10.2	**8**	2.33	**9.8**
`Expedition`	6	9.40	103.7	**8**	**6.21**	**82.7**
`Hydropower`	**9**	7.05	**38.7**	**9**	**3.02**	**38.7**
`Farmland`	10	19.98	250.9	**14**	**7.45**	**237.6**
`Fo-Farmland`	15	25.14	274.6	**16**	**7.58**	**132.1**

Table 1: Comparison between $GBFS$ and $AWBFS$ with the h^n heuristic, in terms of coverage (C), average time (T) in seconds, and average plan length (L). Average time and plan length are evaluated only in the instances solved by both systems.

Conclusion

In this paper, we introduced Adaptive Width Best-First Search (AWBFS), a dynamic parallel search algorithm designed to work with GPU-based parallelization. By adjusting its search width based on heuristic feedback, AWBFS effectively escapes search plateaus while strictly adhering to GPU memory constraints. Our empirical evaluation on IPC 2023 domains demonstrates that AWBFS, when synergistically coupled with batch-processable GNN heuristics and GPU acceleration, outperforms standard GBFS in terms of coverage, runtime, and plan quality.

Acknowledgments

This work has been supported by: MUR PRIN-2020 project RIPER (n. 20203FFYLK). This research was (partly) carried out within the framework of the AI4Water project. The AI4WATER project ("Optimizing Water Resources in Coastal Areas using Artificial Intelligence") is part of the PRIMA Programme supported by the European Union; this project received funding from the Italian Ministry of University and Research (MUR).

References

Bonet, B.; and Geffner, H. 2001. Planning as heuristic search. *Artificial Intelligence*, 129(1-2): 5–33.

Borelli, V.; Gerevini, A.; Scala, E.; and Serina, I. 2026a. Learning Heuristic Functions with Graph Neural Networks for Numeric Planning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, 36172–36179.

Borelli, V.; Gerevini, A. E.; Scala, E.; and Serina, I. 2026b. LeapNP: A Modular Python Framework for Benchmarking Learned Heuristics in Numeric Planning. *Future Internet*, 18(2): 93.

Chen, D.; and Thiébaux, S. 2024. Graph learning for numeric planning. *Advances in Neural Information Processing Systems*, 37: 91156–91183.

Felner, A.; Kraus, S.; and Korf, R. E. 2003. KBFS: K-best-first search. *Annals of Mathematics and Artificial Intelligence*, 39(1): 19–39.

Taitler, A.; Alford, R.; Espasa, J.; Behnke, G.; Fiser, D.; Gimelfarb, M.; Pommerening, F.; Sanner, S.; Scala, E.; Schreiber, D.; Segovia-Aguas, J.; and Seipp, J. 2024. The 2023 International Planning Competition. *AI Mag.*, 45(2): 280–296.