Count Downward Together

Daniel Gnad[1,2], Markus Fritzsche[2], Mikhail Gruntov[3], Alexander Shleyfman[4]

[1] Heidelberg University, Germany
[2] Linköping University, Sweden
[3] Technion – Israel Institute of Technology, Israel
[4] Bar-Ilan University, Israel
daniel.gnad@uni-heidelberg.de, markus.fritzsche@liu.se, gruntovm@campus.technion.ac.il, alexash@biu.ac.il

Abstract

Count Downward Together is a planner for cost-optimal numeric planning built on top of the Numeric Fast Downward framework. Its main contribution is to bring abstraction heuristics, which are among the strongest admissible heuristics in classical planning, to numeric planning tasks. An abstraction maps the concrete state space onto a smaller, abstract one, where optimal goal distances yield admissible estimates. The central obstacle is that projecting a task with numeric variables generally induces an *infinite* abstract state space, which cannot be explored exhaustively. Count Downward Together builds on two recent and complementary families of numeric abstraction heuristics that address this in different ways: numeric pattern-database (PDB) heuristics, which explore only a finite, goal-directed fragment of each abstraction and back it up with admissible estimates on the fringe, and numeric domain-abstraction heuristics, which use counterexample-guided abstraction refinement (CEGAR) to represent the entire infinite concrete state space by partitioning variable domains into intervals. We combine each abstraction collection admissibly via the canonical heuristic, and run these heuristics *together* with numeric LM-cut in sequential portfolios. We submit two portfolios to the optimal track: one running plain A*, and one in which every component additionally exploits task symmetries through orbit-space search.

General Overview

Numeric planning extends classical planning with state variables that take numerical values, allowing one to naturally model resources, capacities, and spatial quantities. This expressiveness comes at a price: numeric tasks induce generally *infinite* state spaces, and the plan-existence problem becomes undecidable, even for highly restricted fragments (Helmert 2002; Gnad et al. 2023). We focus on *simple numeric planning* (SNP), where numeric variables are modified by constant increments/decrements and conditions are linear (in)equalities (Hoffmann 2003; Scala, Haslum, and Thiébaux 2016). Following common practice, our heuristics operate on the equivalent *restricted task* (RT) representation.

In classical planning, abstraction heuristics — most prominently pattern databases (PDBs) and their generalizations — are among the strongest known admissible heuristics (Culberson and Schaeffer 1998; Haslum et al. 2007). An abstraction maps the (concrete) state space onto a smaller, abstract one; optimal goal distances in the abstraction yield an admissible heuristic for the original task. The key difficulty in transferring this idea to numeric planning is that an abstraction that retains even a single numeric variable typically has an infinite abstract state space, so it cannot be explored exhaustively as in the finite-domain case.

Count Downward Together builds on two recent, complementary families of numeric abstraction heuristics that resolve this difficulty in different ways, which form the heuristic core of our optimal-track portfolios. *Numeric PDB heuristics* project the task onto a *pattern*, a subset of the variables (Gnad et al. 2025; Fritzsche et al. 2026a). Since the resulting abstraction may be infinite, only a finite, reachable fragment around the abstract initial state is generated, and admissible goal-distance estimates are attached to the *fringe* of the explored region. Using a blind backup on the fringe, however, often yields uninformative heuristics, in particular when no abstract goal is reached. We therefore use the improved framework of Fritzsche et al. (2026a), which (i) explores the abstraction with a targeted A* search guided by an admissible heuristic so that abstract goal states are actually found, (ii) refines fringe estimates with that same heuristic instead of the minimal action cost, and (iii) provides an admissible fallback for concrete states whose abstract counterpart was never generated (a failed lookup). Multiple small PDBs are generated by a hill-climbing procedure (iPDB) (Haslum et al. 2007) and combined admissibly with the canonical heuristic h^C.

Numeric domain-abstraction heuristics take a different route to the same infiniteness problem (Fritzsche et al. 2026b). A domain abstraction partitions the domain of each variable into blocks; for numeric variables these blocks are integer intervals that may be unbounded, so a *single* abstraction can represent the entire infinite concrete state space without enumerating it. The abstractions are built incrementally by counterexample-guided abstraction refinement (CEGAR) (Clarke et al. 2000; Seipp and Helmert 2018; Kreft et al. 2023): an optimal abstract plan is repeatedly checked against the concrete task, and the discovered precondition, goal, or deviation flaws are eliminated by splitting an interval at a concrete value. As for PDBs, multiple abstractions are combined admissibly using the canonical heuristic h^C.

These two heuristic families are complementary in their search guidance, and each is somewhat competitive with the strongest admissible numeric heuristics, such as numeric

Portfolio	Search	Components (run in sequence)
1	A*	dom. abstractions; PDBs; LM-cut
2	orbit A*	dom. abstractions; PDBs; LM-cut

Table 1: The two optimal-track portfolios. Both run the same three components in sequence; they differ only in the search algorithm, with portfolio 2 adding symmetry-based state pruning via orbit-space search.

LM-cut (Kuroiwa et al. 2022), on a substantial fraction of common SNP benchmarks. As neither dominates the others, we do not commit to a single heuristic. Instead, we run all three *together* in a sequential portfolio, executing the component searches one after another, each on an equal share of the time budget.

We submit two such portfolios to the optimal track. The second is identical to the first, except that every component performs orbit-space search, which detects structural symmetries of the planning task and prunes symmetric states during search (Shleyfman, Kuroiwa, and Beck 2023). This pruning is orthogonal to the heuristic and preserves optimality, so it can further reduce the explored state space.

Implementation & Configurations

Count Downward Together is implemented as an extension of the Numeric Fast Downward (NFD) framework (Aldinger and Nebel 2017), an extension of Fast Downward (Helmert 2006) to numeric state variables. We use NFD's translator to obtain a finite-domain encoding of the numeric task, which our abstraction heuristics transform into the restricted-task (RT) formalism that they operate on (Gnad et al. 2025). We participate only in the optimal track, to which we submit the two sequential portfolios summarized in Table 1. Each portfolio runs three A* configurations one after another, each allotted an equal share of the overall time budget. All configurations target the SNP fragment.

Portfolio Components. The three components differ in the admissible heuristic that guides A*. The first uses *numeric domain-abstraction heuristics* generated by CEGAR (Fritzsche et al. 2026b), combined admissibly with the canonical heuristic h^C. We refine abstractions for up to 300 seconds, bound a single abstraction by $5 \cdot 10^5$ and the collection by $5 \cdot 10^6$ abstract states, and split numeric intervals with the best splitting strategy reported by Fritzsche et al. (2026b). We diversify the collection through a blacklisting mechanism (Rovner, Sievers, and Helmert 2019; Kreft et al. 2023) that excludes a random subset of variables from refinement once 60% of the time has elapsed. The second uses *numeric PDB heuristics* (Gnad et al. 2025; Fritzsche et al. 2026a), again combined with the canonical heuristic h^C. The pattern collection is generated by hill climbing (iPDB) (Haslum et al. 2007) for up to 300 seconds, with up to 10^4 abstract states per pattern and 10^6 as the estimated size bound on a single PDB and on the whole collection. We use the numeric LM-cut heuristic (Kuroiwa et al. 2022) both to guide the goal-directed exploration of each abstraction and to estimate goal distances on its fringe, and fall back to a blind estimate for concrete states whose abstract counterpart was not generated; in the terminology of Fritzsche et al. (2026a), this is the LLB instantiation. The third component runs the numeric LM-cut heuristic (Kuroiwa et al. 2022) directly.

Symmetry-Based Pruning. The two portfolios differ only in the search algorithm. The first runs standard A*, the second replaces it by orbit-space search, which detects structural symmetries of the task and prunes symmetric states during the search (Shleyfman, Kuroiwa, and Beck 2023). This pruning is orthogonal to the heuristic, so it applies uniformly to all three components and can yield further reductions of the explored state space.

Acknowledgments

As we build on (Numeric) Fast Downward, we would like to thank all of its contributors. A big thank you also goes to the organizers of the numeric tracks of the 2026 International Planning Competition.

References

Aldinger, J.; and Nebel, B. 2017. Interval Based Relaxation Heuristics for Numeric Planning with Action Costs. In Kern-Isberner, G.; Fürnkranz, J.; and Thimm, M., eds., *Proceedings of the 40th Annual German Conference on Artificial Intelligence (KI 2017)*, volume 10505 of *Lecture Notes in Artificial Intelligence*, 15–28. Springer-Verlag.

Clarke, E. M.; Grumberg, O.; Jha, S.; Lu, Y.; and Veith, H. 2000. Counterexample-Guided Abstraction Refinement. In Emerson, E. A.; and Sistla, A. P., eds., *Proceedings of the 12th International Conference on Computer Aided Verification (CAV 2000)*, 154–169.

Culberson, J. C.; and Schaeffer, J. 1998. Pattern Databases. *Computational Intelligence*, 14(3): 318–334.

Fritzsche, M.; Gnad, D.; Gruntov, M.; and Shleyfman, A. 2026a. Managing Infinite Abstractions in Numeric Pattern Database Heuristics. In Jenkins, C.; and Taylor, M., eds., *Proceedings of the Fortieth AAAI Conference on Artificial Intelligence (AAAI 2026)*. AAAI Press.

Fritzsche, M.; Gruntov, M.; Shleyfman, A.; and Gnad, D. 2026b. Domain-Abstraction Heuristics for Simple Numeric Planning. In *Proceedings of the 19th Annual Symposium on Combinatorial Search (SoCS 2026)*. AAAI Press. Accepted.

Gnad, D.; Alon, L.; Weiss, E.; and Shleyfman, A. 2025. PDBs Go Numeric: Pattern-Database Heuristics for Simple Numeric Planning. In Shah, J.; and Kolter, Z., eds., *Proceedings of the Thirty-Ninth AAAI Conference on Artificial Intelligence (AAAI 2025)*, 26507–26515. AAAI Press.

Gnad, D.; Helmert, M.; Jonsson, P.; and Shleyfman, A. 2023. Planning over Integers: Compilations and Undecidability. In Koenig, S.; Stern, R.; and Vallati, M., eds., *Proceedings of the Thirty-Third International Conference on Automated Planning and Scheduling (ICAPS 2023)*, 148–152. AAAI Press.

Haslum, P.; Botea, A.; Helmert, M.; Bonet, B.; and Koenig, S. 2007. Domain-Independent Construction of Pattern

Database Heuristics for Cost-Optimal Planning. In *Proceedings of the Twenty-Second AAAI Conference on Artificial Intelligence (AAAI 2007)*, 1007–1012. AAAI Press.

Helmert, M. 2002. Decidability and Undecidability Results for Planning with Numerical State Variables. In Ghallab, M.; Hertzberg, J.; and Traverso, P., eds., *Proceedings of the Sixth International Conference on Artificial Intelligence Planning and Scheduling (AIPS 2002)*, 303–312. AAAI Press.

Helmert, M. 2006. The Fast Downward Planning System. *Journal of Artificial Intelligence Research*, 26: 191–246.

Hoffmann, J. 2003. The Metric-FF Planning System: Translating 'Ignoring Delete Lists' to Numeric State Variables. *Journal of Artificial Intelligence Research*, 20: 291–341.

Kreft, R.; Büchner, C.; Sievers, S.; and Helmert, M. 2023. Computing Domain Abstractions for Optimal Classical Planning with Counterexample-Guided Abstraction Refinement. In Koenig, S.; Stern, R.; and Vallati, M., eds., *Proceedings of the Thirty-Third International Conference on Automated Planning and Scheduling (ICAPS 2023)*, 221–226. AAAI Press.

Kuroiwa, R.; Shleyfman, A.; Piacentini, C.; Castro, M. P.; and Beck, J. C. 2022. The LM-Cut Heuristic Family for Optimal Numeric Planning with Simple Conditions. *Journal of Artificial Intelligence Research*, 75: 1477–1548.

Rovner, A.; Sievers, S.; and Helmert, M. 2019. Counterexample-Guided Abstraction Refinement for Pattern Selection in Optimal Classical Planning. In Lipovetzky, N.; Onaindia, E.; and Smith, D. E., eds., *Proceedings of the Twenty-Ninth International Conference on Automated Planning and Scheduling (ICAPS 2019)*, 362–367. AAAI Press.

Scala, E.; Haslum, P.; and Thiébaux, S. 2016. Heuristics for Numeric Planning via Subgoaling. In Kambhampati, S., ed., *Proceedings of the 25th International Joint Conference on Artificial Intelligence (IJCAI 2016)*, 3228–3234. AAAI Press.

Seipp, J.; and Helmert, M. 2018. Counterexample-Guided Cartesian Abstraction Refinement for Classical Planning. *Journal of Artificial Intelligence Research*, 62: 535–577.

Shleyfman, A.; Kuroiwa, R.; and Beck, J. C. 2023. Symmetry Detection and Breaking in Linear Cost-Optimal Numeric Planning. In Koenig, S.; Stern, R.; and Vallati, M., eds., *Proceedings of the Thirty-Third International Conference on Automated Planning and Scheduling (ICAPS 2023)*, 393–401. AAAI Press.