Tempest Numeric: A Planner for the IPC 2026 Numeric Track

Josef Bajada

Department of Artificial Intelligence, University of Malta
Msida, Malta
josef.bajada@um.edu.mt

Abstract

Tempest is a heuristic forward-search planner for classical and numeric PDDL. It pairs a Metric-FF-style relaxed-planning-graph heuristic that uses interval-based numeric relaxation, with a relaxed planning technique that takes into account numeric supporting effects and supporting action multiplicity. The heuristic also accounts for resource consumption and effects where the amount is not constant, yielding a more informed estimate under certain conditions.

Overview

Tempest is a forward state-space planner supporting PDDL2.1 (Fox and Long 2003). While the fully-fledged planner is designed as a full temporal-numeric planner, the version submitted for the International Planning Competition (IPC) 2026 – Numeric Track is restricted to only propositional and numeric capabilities. In the rest of this paper, we refer to this specific flavour of the planner as Tempest Numeric, to disambiguate from the full planner.

In this debut submission, we targeted only the Simple Numeric Planning (SNP) fragments in the satisficing track. Tempest Numeric supports the instantaneous fragment of numeric PDDL including STRIPS with typing, negative preconditions, disjunctive preconditions and goals, equality, and the numeric-fluents feature set: numeric comparisons in preconditions and goals, the `assign`, `increase`, `decrease`, `scale-up` and `scale-down` effects, and arbitrary arithmetic expressions, including products and quotients of fluents. The planner was implemented from scratch, using Rust, with no external planning dependencies.

Preprocessing and Grounding

Tempest Numeric has an efficient preprocessing and grounding pipeline, designed to prune out fluents and actions to speed up the search and reduce the dimensionality and branching factor of the search space. The normalisation steps include:

Precondition normalisation. Conventional normalisation to Negation Normal Form (NNF) and Disjunctive Normal Form (DNF) with action cloning for each disjunct (Gazen and Knoblock 1997).

Positive normal form. Negative literals are mirrored as positive predicates (Helmert 2006), so that the delete-relaxation heuristic loses no information for them.

Numeric expression normalisation. Every numeric expression in a condition or effect is rewritten to a canonical polynomial form. Constants are folded, coefficients of identical variable products collected, factors and monomials are sorted, and constant denominators are divided into their coefficients.

The grounding process makes use of a static-fluent analysis of the initial state to identify the fluents that no action ever mutates. This is used to restrict the parameter-binding combinations and discard actions whose conditions cannot hold. The relaxed planning graph computed from the initial state is then used to eliminate any actions that are unreachable under relaxation or irrelevant to the goal (Nebel, Dimopoulos, and Koehler 1997).

Before and after grounding, Tempest Numeric runs a domain optimisation pass that rewrites actions in place and discards those it can prove inapplicable. Static fluents identified by the static-fluent analysis (Helmert 2006) are substituted by their initial values throughout preconditions and effects, after which the numeric expression normaliser folds the resulting constants. A numeric effect that collapses to a no-op under this substitution — for instance an `increase` whose rate has reduced to zero — is removed.

The pass also performs a sound bounds analysis over the initial state. For each propositional atom, it determines whether any action can add or delete it. For each numeric fluent, rather than computing its full reachable interval (Scala et al. 2016), it records only the direction in which actions can move it — whether any action can drive it above or below its initial value. From these monotonicity directions it detects conditions that can never hold, which are reduced to falsity, and the actions that carry them are pruned. Since eliminating effects or actions can turn a previously dynamic fluent into a static one, and thereby unlock further optimisations, the domain analysis is recomputed until no further structural changes take place.

Optimising the lifted domain first shrinks the parameter-binding combinations the grounder must enumerate, while the post-grounding optimisation removes the residual ground actions whose conditions become unsatisfiable once their parameters are bound.

Three structural analyses are then computed once over the initial state: per-action execution bounds (used by the heuristic), disjunctive action landmarks from the initial relaxed planning graph (used by the A* fallback), and the macro-action analysis (used by the successor generator).

Numeric Relaxed-Plan Heuristic

The guidance heuristic is a relaxed-planning-graph (RPG) heuristic in the style of FF (Hoffmann and Nebel 2001) and Metric-FF (Hoffmann 2003). An alternating fact/action layered graph is built from the current state under the delete-relaxation; after the goal is reached, a relaxed plan is extracted by regressing goals top-down and recursively adding the preconditions of the supporting actions. The heuristic value h_{FF} is the size of the extracted relaxed plan, and the *helpful actions* are the relaxed-plan actions that are already applicable in the current state. Conversely, when the layered graph saturates without ever reaching the goal, no relaxed plan exists and h_{FF} is infinite. Since the relaxation is a necessary condition for reachability, such a state is a guaranteed dead-end, and the search prunes it rather than expanding its successors.

Numeric Relaxation and Supporting Effects

In the relaxed graph each numeric fluent is tracked as a closed interval $[lo, hi]$ with possibly unbounded ends, which can only grow as layers are expanded. Per-action execution bounds determine how far a numeric effect is propagated within a layer. These bounds are computed from single-shot and k-shot action analysis, which determine the upper bound on how many times an action can be executed (Coles et al. 2010; Benton, Coles, and Coles 2012). These are derived by a fixpoint over the initial state: an action that deletes a fact no other action can re-establish is single-shot ($k = 1$), and an action that consumes a numeric resource no other action replenishes is k-shot for the k its supply allows. A bounded action then contributes the cumulative effect of up to its maximum number of applications, while an unbounded action extrapolates the fluent either towards $\pm\infty$, or asymptotically towards 0 for scaling effects with a factor between -1 and 1. Effect amounts are themselves evaluated with interval arithmetic over the relaxed state, so an effect whose magnitude depends on other fluents widens its target correctly, propagating the effects across multiple fluents (Bajada, Fox, and Long 2015, 2016). A numeric goal or precondition is then satisfied in the relaxation as soon as the relevant interval admits a satisfying value. Interval endpoints are represented as exact arbitrary-precision decimals, so neither the relaxation nor actual action application gains or loses precision through floating-point rounding, which could otherwise accumulate significant error across long effect chains.

Relaxed-Plan Extraction

When the extracted plan supports a numeric condition, the supporting action is counted with a *multiplicity* factor rather than just once (Coles et al. 2008). The planner estimates the number of action applications needed to close the gap (Scala, Haslum, and Thiébaux 2016). Action multiplicities enable further refinements:

- Consumers of a numeric fluent raise the required multiplicity of its producers (demand propagation).
- Producers are added when the plan consumes more of a resource than the state holds (potentially resulting in a dead-end if no such producers are applicable).
- Equality goals are handled as a special case, when multiple producer/consumer interactions are needed to arrive at the required value.

Tempest Numeric also handles numeric effects whose amount is not constant — for example an increase whose rate is itself a fluent. This follows the same *constants in context* treatment that was originally implemented for continuous effects (Bajada, Fox, and Long 2016, 2015, 2022), but in this case adapted for discrete numeric effects. It derives an implicit sub-goal that the effect's amount must carry to move another fluent towards the parent goal. This pulls the action that modifies the dependent fluent into the relaxed plan, so it surfaces as a helpful action.

Search

The planner supports a number of configurable search strategies, including Greedy Best-First Search (GBFS) (Bonet and Geffner 2001), Enforced Hill Climbing (EHC) (Hoffmann and Nebel 2001) and A* search (Hart, Nilsson, and Raphael 1968), but the configuration we submit for the SNP-SAT track makes use of EHC as the primary search, with a complete A* search as fallback. While the original EHC does not order the helpful-action successors, we use a min-heap ordering on the heuristic value, with ties broken by insertion order for deterministic behaviour.

Because EHC over helpful actions is incomplete, on failure the planner falls back to a complete A* search over all applicable actions, using the same heuristic, but also consulting disjunctive action landmarks (not used in the EHC stage). Each node in the fallback search carries a bitset of landmarks not yet achieved, and its estimate is combined with the landmark count (Hoffmann, Porteous, and Sebastia 2004; Keyder, Richter, and Helmert 2010).

Macro-Action Successors

On domains where the relaxed plan indicates that a single action must fire many times in succession (for example Manhattan-distance-style movement), Tempest Numeric folds runs of identical applications into multi-step macro edges (Botea et al. 2005). An action is eligible when its only effect is a constant increase or decrease of one numeric fluent with no propositional change. The successor generator then emits one edge per relevant multiplicity, that is, when values are aligned with goal and precondition thresholds. This macro mechanism is enabled automatically only when some action's multiplicity in the initial relaxed plan reaches a small threshold, so that resource-flow domains, where each action fires only once or twice, are unaffected. Plan output always remains unit-action, and macro edges are expanded back into their underlying action sequence.

References

Bajada, J.; Fox, M.; and Long, D. 2015. Temporal Planning with Semantic Attachment of Non-Linear Monotonic Continuous Behaviours. In *Proceedings of the 24th International Joint Conference on Artificial Intelligence (IJCAI)*, 1523–1529.

Bajada, J.; Fox, M.; and Long, D. 2016. Temporal Planning with Constants in Context. In *Proceedings of the 22nd European Conference on Artificial Intelligence (ECAI)*, volume 285 of *Frontiers in Artificial Intelligence and Applications*, 1712–1713. IOS Press.

Bajada, J.; Fox, M.; and Long, D. 2022. Efficient Temporal Piecewise-Linear Numeric Planning with Lazy Consistency Checking. *IEEE Transactions on Artificial Intelligence*, 3(4): 506–517.

Benton, J.; Coles, A. J.; and Coles, A. I. 2012. Temporal Planning with Preferences and Time-Dependent Continuous Costs. In *Proceedings of the 22nd International Conference on Automated Planning and Scheduling (ICAPS)*, 2–10. AAAI Press.

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

Botea, A.; Enzenberger, M.; Müller, M.; and Schaeffer, J. 2005. Macro-FF: Improving AI Planning with Automatically Learned Macro-Operators. *Journal of Artificial Intelligence Research*, 24: 581–621.

Coles, A.; Fox, M.; Long, D.; and Smith, A. 2008. A Hybrid Relaxed Planning Graph-LP Heuristic for Numeric Planning Domains. In *Proceedings of the 18th International Conference on Automated Planning and Scheduling (ICAPS)*, 52–59. AAAI Press.

Coles, A. J.; Coles, A. I.; Fox, M.; and Long, D. 2010. Forward-Chaining Partial-Order Planning. In *Proceedings of the 20th International Conference on Automated Planning and Scheduling (ICAPS)*, 42–49. AAAI Press.

Fox, M.; and Long, D. 2003. PDDL2.1: An Extension to PDDL for Expressing Temporal Planning Domains. *Journal of Artificial Intelligence Research*, 20: 61–124.

Gazen, B. C.; and Knoblock, C. A. 1997. Combining the Expressivity of UCPOP with the Efficiency of Graphplan. In *Recent Advances in AI Planning (ECP-97)*, volume 1348 of *LNAI*, 221–233. Springer.

Hart, P. E.; Nilsson, N. J.; and Raphael, B. 1968. A Formal Basis for the Heuristic Determination of Minimum Cost Paths. *IEEE Transactions on Systems Science and Cybernetics*, 4(2): 100–107.

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.

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

Hoffmann, J.; Porteous, J.; and Sebastia, L. 2004. Ordered Landmarks in Planning. *Journal of Artificial Intelligence Research*, 22: 215–278.

Keyder, E.; Richter, S.; and Helmert, M. 2010. Sound and Complete Landmarks for And/Or Graphs. In *Proceedings of the 19th European Conference on Artificial Intelligence (ECAI)*, volume 215 of *Frontiers in Artificial Intelligence and Applications*, 335–340. IOS Press.

Nebel, B.; Dimopoulos, Y.; and Koehler, J. 1997. Ignoring Irrelevant Facts and Operators in Plan Generation. In *Recent Advances in AI Planning (ECP-97)*, volume 1348 of *LNAI*, 338–350. Springer.

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

Scala, E.; Haslum, P.; Thiébaux, S.; and Ramirez, M. 2016. Interval-Based Relaxation for General Numeric Planning. In *Proceedings of the 22nd European Conference on Artificial Intelligence (ECAI)*, volume 285 of *Frontiers in Artificial Intelligence and Applications*, 655–663. IOS Press.