Interval Based Relaxation Heuristics for Numeric Planning with Action Costs*

Johannes Aldinger, Bernhard Nebel
University of Freiburg, Department of Computer Science
D-79110 Freiburg, Germany
{aldinger, nebel}@informatik.uni-freiburg.de

Abstract

We adapt the relaxation heuristics $h_{\max}$, h_{add} and h_{FF} to interval based numeric relaxation frameworks, combining them with two different relaxation techniques and with two different search techniques. In contrast to previous approaches, the heuristics presented here are not limited to a subset of numeric planning and support action costs.

Numeric planning allows to model numeric quantities such as physical properties (e.g. velocity) and resources (e.g. fuel level) in addition to the propositional properties of classical planning. Numeric planning with instantaneous actions is expressible with PDDL2.1, layer 2 (Fox and Long 2003).

We investigate adaptations of the delete relaxation heuristics $h_{\max}$, h_{add} and h_{FF} to numeric planning. In contrast to other state-of-the art approaches (Hoffmann 2003; Scala, Haslum, and Thiébaux 2016; Scala et al. 2016) we are interested in adaptations which offer heuristic guidance for *all* numeric planning tasks including actions with non-linear effects and non-uniform action cost. Notably, we are also interested in computing h_{FF}, a heuristic basing its estimate on the extraction of valid relaxed plans.

Tractable heuristics are polynomial-time in either the input or, at least, in the output size. A challenge of numeric planning are non-idempotent actions: applying the same numeric effect to a state more than once can yield a new successor every time. Therefore, *interval relaxation* based heuristics are only polynomial in the length of a shortest relaxed plan (Hoffmann 2003). Recently, Aldinger et al. Aldinger, Mattmüller, and Göbelbecker (2015) proposed an interval-based relaxation framework for numeric planning that captures arbitrary many applications of a numeric action in one step. The plan existence problem in this *repetition relaxation* is polynomial in the input for acyclic tasks.

We compare the two relaxation methods (*interval* or *repetition* relaxation) with the *planning graph method* and *priority queues* as search techniques for relaxed reachability. Finally, we present a generalization to the marking method of relevant operators used by h_{FF}, which explicates target values in the intervals to extract relaxed plans.

*This work was supported by the DFG grant EXC1086 BrainLinks-BrainTools to the University of Freiburg, Germany.

Numeric Relaxation Heuristics

The relaxation heuristics $h_{\max}$, h_{add} estimate the cost $\gamma(v)$ to achieve the propositions v for each achieving action a by $\gamma(v) := \min_a(\gamma(v), \gamma(a) + \gamma(\text{pre}(a)))$, where $\gamma(\text{pre}(a))$ is a cost estimate of a set of propositions, which is the most expensive proposition cost $\gamma(\text{pre}(a)) := \max_{v \in \text{pre}(a)} \gamma(v)$ for $h_{\max}$, and the *sum* of all preconditions $\gamma(\text{pre}(a)) := \sum_{v \in \text{pre}(a)} \gamma(v)$ for h_{add}.

In numeric planning, we have to assign costs estimates to variable-value pairs as well. These *numeric facts* are *implicit preconditions* on variables that appear in the expressions ξ of explicit preconditions $\xi \trianglerighteq 0$ or of numeric effects $v \circ\!= \xi$. Numeric relaxation heuristics have to consider a bounded number of numeric facts which ensure that values in the precondition enable the required values in the effect. To restrict this number, we discuss a *planning graph* based and a *priority queue* based approach. Both are motivated by ways of computing the cost formulas in classical planning.

A relaxed *planning graph* restricts the number of considered numeric facts to one variable-value pair for each variable in the state layers of the graph. For interval based planning, several parallel actions can alter the same variable, in which case the convex union of the individual results is used. The length of a shortest relaxed plan restricts the maximal number of layers required until the goal formula is satisfied for the first time, thus bounding the number of considered facts polynomially in the output. While most state-of-the art planners use this approach, *interval relaxation* based heuristics can overestimate the cost of numeric facts, making $h_{\max}$ inadmissible. The reason is that a fact can be achieved at a better cost in a deeper layer in the planning graph. For numeric planning, no a priori bound can be given on how deep the better value could be found. Opposed to relaxed classical planning, *interval relaxed* facts will usually not reach a fixpoint, and their cost is determined at their first occurrence.

A different approach to restrict the considered numeric facts is to use a generalized Dijkstra algorithm for estimating the fact or fact set costs. Facts are processed according to a *priority queue* storing the cost to achieve them. As other actions can alter a variable while a new value is still in the priority queue, the considered numeric facts are convex unions of the effects values reachable at enqueue time and the variable's value at dequeue time. The approach is incompatible with *interval relaxation* based heuristics as unboundedly

many cheap actions could be processed before relevant ones. Even *repetition relaxed* actions are not entirely idempotent. The result of a numeric effect *depends* on the variables in the assigned expression. If the induced dependency graph is acyclic, variables reach a fix-point in order of the topology of this graph. However, if cheap actions depend on many other more expensive actions which reside in topologically higher layers, the topologically lower variables could be enqueued exponentially more often. The heuristic becomes tractable if the actions which have an *implicit precondition* on a variable are only enqueued after the changes triggered from topologically higher variables have been processed. This means that variables in the lower layers have to wait for variables in a higher layer even though they could be more expensive. As the values achieved by the topologically higher variables might not be required, the cost can be overestimated. While this is acceptable for h_{add}, $h_{\max}$ requires admissibility. Thus, we compute $h_{\max}$ in two phases: we determine maximally reachable intervals with topology blocking in the first phase and use these maximally reachable intervals in the second phase, where blocking is not required, as now action sequences are idempotent. In the presence of cycles in the *dependency graph* we introduce auxiliary variables for variables that appear in a cycle. Cycle breaker actions can then reinsert the values in a controlled manner, ensuring that running through the cycle becomes pseudo-idempotent.

The h_{FF} heuristic computes a *relaxed plan*, and uses the cost of this plan as heuristic estimate. As in classical planning, this plan is computed *regressively* by *marking* required facts and actions based on the h_{add} estimate. Numeric planning can improve upon h_{add} even more than in classical planning by not having to fully enable *implicit preconditions*, as numeric facts do not necessarily have to enable the whole reachable interval. Therefore, marking a numeric fact involves explicating a *target values* in these intervals.

The progression step generates a monotonically increasing sequence of relaxed states that starts with point intervals and satisfies the goal condition in the end. We want to explicate target values in these intervals so that the resulting plan has minimal cost. The explication procedure has to respect *local target value constraints* for each action, which ensure that all implicit and explicit preconditions of the action are satisfied and that the action achieves the desired values. These *local target value constraints* generate a set of feasible sub-intervals, where each choice of an explicit target value can lead to some relaxed plan.

The *global target value optimization problem* is then an optimization problem that selects target values in the feasible sub-intervals which minimize the cost of the resulting relaxed plan. Each target value choice influences the *local target value constraints* of all preceding states.

The *local target value constraints* ensure that for each numeric effect of an action, all implicit and explicit preconditions are satisfied in the relaxed state before the application of the action, and that the execution of its numeric effects enables the target values desired from the global optimization component. Three basic target value conditions ensure satisfaction a *local target value constraint*: Its explicit preconditions have to be satisfied (1). This requires numeric expressions evaluate to a desired target value (2). Finally, the numeric effects have to reach the desired target value (3). These basic target value conditions then restrict the intervals of the preceding state by sub-intervals containing feasible selection choices for the global target value optimization.

The target value explication process selects locally promising target values. The values of all variables have to reach the point intervals at the end of the regression procedure, making proximity to these starting values an indicator for good target values. An exception to this rule occurs are open intervals in the *repetition relaxation*. As open bounds are only generated by contracting effects, values close to open bounds can only be reached by applying the contraction repeatedly and it is advisable to keep a safety margin to open interval bounds.

Experiments

We implemented a numeric extension (NFD) for the Fast Downward planning system (Helmert 2006). We show experiments for h_{FF} in the two most promising combinations of relaxation and "fact" selection scheme identified in the previous section: the *planning graph* approach in an *interval relaxation* h^{gi} and the *priority queue* based approach in the *repetition relaxation* h^{qr}. We performed experiments on various numeric domains comparing NFD with Metric FF (MFF) (Hoffmann 2003) and two configurations of ENHSP: subgoaling with redundant constraints $\hat{h}^{radd}_{hbd+}$ (Scala, Haslum, and Thiébaux 2016) and h_{AIBR} (Scala et al. 2016). Our heuristics perform similar to Metric FF for the planning domains that both planners can solve. Metric FF is restricted to linear tasks and cannot find solutions for non-linear cyclic problems such as jumpbot. ENHSP performs often better, but it ignores action costs. There were no domain in the benchmarks that exploits this weakness. Table 1 shows the solved instances on selected numeric domains.

	MFF	ENHSP		NFD	
		$\hat{h}^{radd}_{hbd+}$	h_{AIBR}	h^{gi}_{FF}	h^{qr}_{FF}
counters (34)	**7**	2	4	**7**	**6**
jumpbot (20)	0	3	**15**	**15**	8
settlers (20)	**9**	0	0	0	3

Table 1: Solved instances on selected numeric domains

References

Aldinger, J.; Mattmüller, R.; and Göbelbecker, M. 2015. Complexity of Interval Relaxed Numeric Planning. In *KI 2015*.

Fox, M., and Long, D. 2003. `PDDL2.1` : An Extension to `PDDL` for Expressing Temporal Planning Domains. *JAIR 2003*.

Helmert, M. 2006. The Fast Downward Planning System. *JAIR 2006*.

Hoffmann, J. 2003. The Metric-FF Planning System: Translating 'Ignoring Delete Lists' to Numeric State Variables. *JAIR 2003*.

Scala, E.; Haslum, P.; Thiébaux, S.; and Ramírez, M. 2016. Interval-Based Relaxation for General Numeric Planning. In *ECAI 2016*.

Scala, E.; Haslum, P.; and Thiébaux, S. 2016. Heuristics for Numeric Planning via Subgoaling. In *IJCAI 2016*.