Delete Relaxations for Planning with State-Dependent Action Costs

Florian Geißer and **Thomas Keller** and **Robert Mattmüller**
University of Freiburg, Germany
{geisserf,tkeller,mattmuel}@informatik.uni-freiburg.de

Abstract

Supporting state-dependent action costs in planning admits a more compact representation of many tasks. We generalize the additive heuristic h^{add} and compute it by embedding decision-diagram representations of action cost functions into the RPG. We give a theoretical evaluation and present an implementation of the generalized h^{add} heuristic. This allows us to handle even the hardest instances of the combinatorial ACADEMIC ADVISING domain from the IPPC 2014.[1]

Introduction and Preliminaries

State-dependent action costs (SDAC) often admit more compact domain representations than state-independent action costs. We show how action cost functions can be represented as *edge-valued multi-valued decision diagrams (EVMDDs)* (Ciardo and Siminiceanu 2002), which allows us to *detect and exploit structure* in the cost functions. EVMDDs can be used to derive a compact compilation to constant-cost actions or can directly be embedded into the relaxed planning graph (RPG). We define a natural generalization of the additive heuristic h^{add} to SDAC and show that the EVMDD embedding can be used to compute this generalized h^{add} heuristic. We apply this procedure to the ACADEMIC ADVISING domain (Guerin et al. 2012) and compare the results with the standard heuristic of the PROST planner (Keller and Eyerich 2012). We consider SAS^+ tasks (Bäckström and Nebel 1995) with the usual syntax and semantics. Additionally, each action a has an associated *cost function* $c_a : \mathcal{D}_1 \times \cdots \times \mathcal{D}_n \to \mathbb{N}$, where $\mathcal{D}_1, \ldots, \mathcal{D}_n$ are the finite domains of the variables on which c_a depends. Let S_a be the set of valuations of those variables and let F_a be the set of facts (v, d) for $v \in S_a$ and d in the domain of v. Without loss of generality, we assume that for each action a, the sets of variables mentioned in the precondition and those on which c_a depends are disjoint. We may also view c_a as a function over states. We use the usual definition of relaxed planning tasks and define the cost of an action a in a relaxed state s^+, $c_a(s^+)$ as the *minimal* cost $c_a(s)$ for any unrelaxed state s that is subsumed by s^+. Next, we give a generalization of the additive heuristic h^{add} (Bonet and Geffner 2001) to tasks with SDAC that correctly reflects action costs. Let $s_\star$ be the goal description, let s_p stand for a partial state and $f = (v, d)$ for a fact, and let $A(f)$ be the set of *achievers* of f. As in the classical setting, $h^{add}(s) = h^{add}_s(s_\star)$, and $h^{add}_s(s_\mathrm{p}) = \sum_{f \in s_\mathrm{p}} h^{add}_s(f)$. Unlike in the classical setting,

$$h^{add}_s(f) = \begin{cases} 0 & \text{if } f \in s \\ \min\limits_{a \in A(f)} \left[h^{add}_s(\mathit{pre}(a)) + C^a_s\right] & \text{else,} \end{cases}$$

$$\text{where } C^a_s = \min_{\hat{s} \in S_a} [c_a(\hat{s}) + h^{add}_s(\hat{s})],$$

where $\mathit{pre}(a)$ is the precondition of a. We minimize over all achievers a of f *and over all possible situations where a is applicable* by replacing the constant action cost of the classical setting with C^a_s. The central new challenge is the *efficient computation* of C^a_s. The number of states in S_a is exponential in the number of variables on which c_a depends, and C^a_s cannot always be additively decomposed by these variables. However, we can represent c_a and C^a_s using EVMDDs such that c_a and C^a_s are efficiently computable in the size of the EVMDD representation and that the size of the representation itself is, although worst-case exponential, compact in many "typical", well-behaved cases.

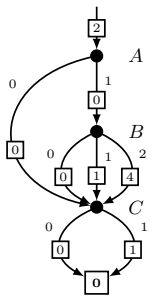
Edge-Valued Decision Diagrams

Each cost function $c_a : \mathcal{D}_1 \times \cdots \times \mathcal{D}_n \to \mathbb{N}$ over variables $v_1, \ldots, v_n$ with domains $\mathcal{D}_v = \{0, \ldots, |\mathcal{D}_v| - 1\}$ can be encoded as an EVMDD (Ciardo and Siminiceanu 2002). EVMDDs are directed acyclic graphs with unique source edges and sink nodes and with interior nodes branching over the domain values of the variables. Edges have associated weights, and for a concrete valuation s of the variables, the value $\mathcal{E}(s)$ of an EVMDD $\mathcal{E}$ is determined by traversing the unique path in $\mathcal{E}$ corresponding to s and summing up the edge weights along the way.

To compute C^a_s, we need to incorporate h^{add}_s values into $\mathcal{E}_a$. To that end, on each path through $\mathcal{E}_a$, each variable on which c_a depends must be tested. Hence, in the following we assume that $\mathcal{E}_a$ includes branches over all variables on all paths, and call such an $\mathcal{E}_a$ *quasi-reduced*.

[1] A long version of this paper was accepted to IJCAI-15 (Geißer, Keller, and Mattmüller 2015). To cite this work, please cite the IJCAI-15 paper.

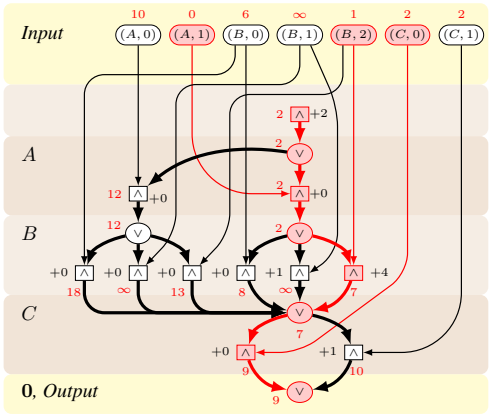
Example 1. *Consider the action cost function* $c_a = AB^2 + C + 2$ *with* $\mathcal{D}_A = \mathcal{D}_C = \{0,1\}$, *and* $\mathcal{D}_B = \{0,1,2\}$. *The figure to the left shows an EVMDD* (not *quasi-reduced) for* c_a *and the variable ordering* A, B, C. *To see how the EVMDD encodes the cost function* c_a, *consider the valuation* s *with* $s(A) = 1$, $s(B) = 2$ *and* $s(C) = 0$. *Traversing the corresponding edges in the EVMDD from top to bottom and summing up the edge weights, we arrive at the resulting value* 6, *which is exactly the cost* $c_a(s)$.



EVMDD-Based RPG Compilation

To embed such an EVMDD $\mathcal{E}_a$ into the RPG, we need to add "input nodes" for all relevant h^{add}_s values to $\mathcal{E}_a$. We call the resulting weighted directed acyclic AND/OR graph $\mathcal{I}_a$ the *input-node augmented decision diagram (INADD)* for $\mathcal{E}_a$. We extend a given valuation ι assigning natural numbers or infinity to input nodes to a valuation $\hat{\iota}$ recursively by minimization at OR nodes and summation plus addition of node weights at AND nodes. The *value* of $\mathcal{I}_a$ for ι, $\mathcal{I}_a(\iota)$, is then the $\hat{\iota}$ value of the terminal/output INADD node.

Example 2. *Consider the EVMDD* $\mathcal{E}_a$ *from Example 1 in its quasi-reduced form* $\mathcal{E}'_a$. *The INADD* $\mathcal{I}_a$ *for* $\mathcal{E}'_a$ *is depicted below. Node weights are given as node annotations.*



The red input node labels denote $\iota(n)$ *for some valuation* ι, *and the red interior node labels denote the valuation* $\hat{\iota}$. *The highlighted red path is the minimizing path determining* $\mathcal{I}_a$, *with used inputs also highlighted. The value* $\mathcal{I}_a = 9$ *equals* $C^a_s = \min_{\hat{s} \in S_a}[c_a(\hat{s}) + h^{add}_s(\hat{s})]$ *if the* ι *values are interpreted as* h^{add}_s *values: The highlighted path corresponds to the minimizing* $\hat{s}$, $c_a(\hat{s})$ *is the sum of the node weights* $2 + 0 + 4 + 0 = 6$ *along the path, and* $h^{add}_s(\hat{s})$ *is the sum of used input node costs,* $0 + 1 + 2 = 3$.

Theorem 1. *Let* $\mathcal{I}_a$ *be the INADD for a quasi-reduced EVMDD* $\mathcal{E}_a$ *encoding the action cost function* c_a *of action* a. *Let the* ι *values of input fact nodes be* $h^{add}_s(f)$ *for all facts* $f \in F_a$. *Then* $\mathcal{I}_a(\iota) = C^a_s$. □

In each RPG layer, we can embed an action sub-graph for a that is the "conjunction" of $\mathcal{I}_a$ with the precondition facts of a. The input nodes of $\mathcal{I}_a$ are the corresponding fact nodes from the previous layer, and the "output" node of $\mathcal{I}_a$ is linked to the effect literals of a on the current layer.

Instance	**2**	**4**	**6**	**8**	**10**
Variables	20	30	40	50	60
Actions	11	16	21	26	31
Variables in c_a	8	8	11	10	12
Size of $\mathcal{I}_a$	$26 + 33$	$26 + 33$	$35 + 45$	$32 + 41$	$38 + 49$
IDS	**46.24**	202.19	202.90	201.67	201.51
h^{add}	**45.80**	**63.41**	**76.15**	**109.02**	**125.52**

Table 1: Statistics of the ACADEMIC ADVISING domain.

Experimental Evaluation

We implemented the generalized h^{add} heuristic in the 2014 version of the PROST planner (Keller and Eyerich 2012) and compared it against the standard heuristic of PROST on the ACADEMIC ADVISING domain. Table 1 shows statistics for different instances. The first four rows show numbers of variables, of actions, of variables occurring in the cost function of each action, and of nodes plus edges of the resulting INADD. The last two rows show the average results of PROST across 100 runs per instance and with a 1-second timeout per planning step, for the standard IDS heuristic used by PROST and for the additive heuristic using the EVMDD-based RPG compilation. Both are applied to the most-likely determinization.

Conclusion

We introduced a natural extension of the definition of the h^{add} heuristic to tasks with SDAC and showed how it can be efficiently computed based on a compact encoding of action cost functions using EVMDDs. We presented an implementation of the generalized h^{add} heuristic in the context of probabilistic planning and showed its efficiency.

Acknowledgements. This work was partly supported by the DFG as part of the SFB/TR 14 AVACS and by BMBF grant 02PJ2667 as part of the KARIS PRO project. We thank the reviewers for their insightful comments.

References

Bäckström, C., and Nebel, B. 1995. Complexity Results for SAS+ Planning. *Computational Intelligence* 11:625–656.

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

Ciardo, G., and Siminiceanu, R. 2002. Using edge-valued decision diagrams for symbolic generation of shortest paths. In *Proc. FMCAD 2002*, 256–273.

Geißer, F.; Keller, T.; and Mattmüller, R. 2015. Delete relaxations for planning with state-dependent action costs. In *Proc. IJCAI 2015*. To appear.

Guerin, J. T.; Hanna, J. P.; Ferland, L.; Mattei, N.; and Goldsmith, J. 2012. The academic advising planning domain. In *Proc. IPC Workshop at ICAPS 2012*.

Keller, T., and Eyerich, P. 2012. PROST: Probabilistic Planning Based on UCT. In *Proc. ICAPS 2012*.