Abstractions += Landmarks

Carmel Domshlak and **Michael Katz** and **Sagi Lefler**
Faculty of Industrial Engineering and Management
Technion—Israel Institute of Technology
Haifa, Israel

Abstract

Abstractions and landmarks are two powerful mechanisms for devising admissible heuristics for classical planning. Here we aim at putting them together by enhancing problem abstractions with landmark information, and propose a concrete realization of this direction suitable for structural-pattern abstractions. Our preliminary evaluation both provides a proof of concept, and suggests directions for further improvements.

Introduction

Heuristic state-space search is a common and successful approach to classical planning, and in particular, to cost-optimal classical planning. Apart from the choice of the search algorithm, heuristic-search solvers for cost-optimal planning differ mainly in their admissible heuristic estimators. Recent years have seen a growing body of work on expanding the palette of heuristic estimators, with most (if not all) current admissible heuristics being based on one of the following three ideas:

1. *critical paths*: the h^m heuristic family (Haslum and Geffner 2000), with the $h^1 \equiv h^{\max}$ member being closely related to the *delete relaxation* idea,
2. *abstractions*: pattern databases (Edelkamp 2001), merge-and-shrink abstractions (Helmert, Haslum, and Hoffmann 2007), and structural patterns (Katz and Domshlak 2008b),
3. *landmarks*: the admissible landmark heuristics h^{L} and h^{LA} (Karpas and Domshlak 2009), closely related to delete relaxation.

Until very recently, these three ideas have been developed in relative isolation, and thus there has been no cross-fertilization between them. In the first work aiming at connecting between the three different approaches, Helmert and Domshak (2009) in particular show that additive $h^{\max}$ and admissible landmark heuristics are in fact very much related. Importantly, this realization allowed the authors to develop a novel admissible landmark heuristic, $h^{\text{LM-cut}}$, that has dramatically changed the state of the art in performance for cost-optimal planning.

In this work we consider another edge of the above triangle of ideas, namely abstractions and landmarks, and try to exploit the best of both worlds by fertilizing the former with the latter. In general, abstraction heuristics have been shown by Helmert and Domshlak (2009) to be more expressive (in a proper sense of this notion) than landmark heuristics. However, all the currently used mechanisms for devising abstraction heuristics have an Achilles heel that, potentially, might be cured by exploiting the information communicated by the problem's landmarks. That Achilles heel is, sometimes direct and sometimes indirect, dependence of the quality of abstraction heuristics on the richness of the goal description that comes with the problem specification. Informally, the fewer the problem sub-goals explicitly mentioned by the problem, the less guided (and thus less effective) are the procedures for selecting concrete sets of abstractions. This is precisely the place where landmarks, constituting *implicit sub-goals* of the problem, have a potential to improve things.

Focusing on fork-decomposition structural patterns (Katz and Domshlak 2008b), we show how landmarks can be exploited in enhancing these abstraction heuristics by compiling the landmarks into the problem specification. Our focus on fork-decomposition is not incidental—among the abstraction heuristics in use to date, these are probably the most sensitive to the richness of the problem's goal specification. The problem compilation we propose is extremely simple, and at the same time preserves all the essential reachability properties of the original problem. Our empirical analysis clearly testifies for the effectiveness of the proposed approach, boosting substantially the quality of the induced heuristic estimates. The results are, how-

ever, only preliminary, and things in practice are still not all that bright. In particular, we show that further investigation of ad hoc action-cost partitioning is required to improve the robustness of the landmark-enhanced abstractions.

Background

We consider problems of classical planning corresponding to state models with single initial state and only deterministic actions; here we consider state models captured by the SAS$^+$ formalism (Bäckström and Nebel 1995) with non-negative action costs. Such a problem is given by a quintuple $\Pi = \langle V, A, I, G, \mathcal{C} \rangle$, where:

- V is a set of *state variables*, each $v \in V$ is associated with a finite domain $dom(v)$; each complete assignment to V is called a *state*, and $S = dom(V)$ is the *state space* of Π. I is an *initial state*. The *goal* G is a partial assignment to V; a state s is a *goal state* iff $G \subset s$.
- A is a finite set of *actions*. Each action a is a pair $\langle \mathsf{pre}(a), \mathsf{eff}(a) \rangle$ of partial assignments to V called *preconditions* and *effects*, respectively. $\mathcal{C} : A \to \mathbb{R}^{0+}$ is a real-valued, non-negative *action cost* function.

The value of a variable v in a partial assignment p is denoted by $p[v]$. By $V(p) \subseteq V$ we denote the set of variables instantiated by p. An action a is applicable in a state s iff $s[v] = \mathsf{pre}(a)[v]$ whenever $\mathsf{pre}(a)[v]$ is specified. Applying a changes the value of v to $\mathsf{eff}(a)[v]$ if $\mathsf{eff}(a)[v]$ is specified. The resulting state is denoted by $s[\![a]\!]$; by $s[\![\langle a_1, \ldots, a_k \rangle]\!]$ we denote the state obtained from sequential application of the (respectively applicable) actions $a_1, \ldots, a_k$ starting at state s. Such an action sequence is an *s-plan* if $G \subseteq s[\![\langle a_1, \ldots, a_k \rangle]\!]$, and it is a *cost-optimal* (or, in what follows, *optimal*) s-plan if the sum of its action costs is minimal among all s-plans. The purpose of (optimal) planning is finding an (optimal) I-plan.

For a pair of states $s_1, s_2 \in S$, by $\mathcal{C}(s_1, s_2)$ we refer to the cost of a cheapest action sequence taking us from s_1 to s_2 in the transition system induced by Π; $h^*(s) = \min_{s' \supseteq G} \mathcal{C}(s, s')$ is the custom notation for the cost of optimal s-plans for Π.

Let $\Pi = \langle V, A, I, G, \mathcal{C} \rangle$ be a planning task with variable domains $dom(v_i)$, $F = \bigcup_i dom(v_i)$ be the set of *facts* (assuming name uniqueness), ϕ be a propositional logic formula over facts F, $\pi = \langle a_1, \ldots, a_k \rangle$ be an action sequence applicable in I, and $0 \leq i \leq k$. Following the terminology of Hoffmann *et al.* 2004, we say that ϕ is *true at time i* in π iff $I[\![\langle a_1, \ldots, a_i \rangle]\!] \models \phi$, and ϕ is a *landmark* of Π iff in each plan for Π, it is true at some time.

While landmarks can be any formulas over facts, we restrict our attention to disjunctions of facts, and use notation $\phi \subseteq F$ to denote "disjunction over the fact subset ϕ of F". This restriction covers all the landmark discovery procedures suggested in the literature. Due to hardness of deciding even that a single fact is a landmark (Porteous, Sebastia, and Hoffmann 2001), practical methods for finding landmarks are either incomplete or unsound. In what follows we assume access to a sound such procedure; in particular, in our evaluation we use LAMA's sound landmark discovery procedure, introduced by Richter *et al.* (2008). The actual way of discovering landmarks is tangential to our work.

Bringing Landmarks into Abstractions

Landmarks are exploited these days in both satisficing and optimal planning as heuristic search, either for devising an incremental, landmark-by-landmark search strategy (Hoffmann, Porteous, and Sebastia 2004) or for deriving heuristic estimates (Richter, Helmert, and Westphal 2008; Karpas and Domshlak 2009; Helmert and Domshlak 2009). In parallel, other sources of information for heuristic guidance have been proven extremely valuable, and this in particular so with various problem abstractions. We now proceed with, first, arguing that landmarks have a natural potential to enhance abstraction heuristics by targeting one of the major sources of their vulnerability. We then describe a simple technique for enhancing structural-pattern abstractions with landmark information, and evaluate and discuss two ways of exploiting this enhancement in heuristic-search optimal planning. In particular, our preliminary empirical evaluation of the framework testifies for the success of the proof of concept.

Abstractions and Goal Specification

An *abstraction* heuristic is based on mapping Π's transition system over states S to an *abstract transition system* over states S^α. The mapping is defined by an abstraction function $\alpha : S \to S^\alpha$ that guarantees $\mathcal{C}_\alpha(\alpha(s), \alpha(s')) \leq \mathcal{C}(s, s')$ for all states $s, s' \in S$. The abstraction heuristic $h^\alpha(s)$ is then the distance from $\alpha(s)$ to the closest abstract goal state. Abstraction heuristics are always admissible by their very definition. Two families of abstractions are used these days for deriving admissible heuristics. Abstractions such as in pattern databases (Edelkamp 2001; Haslum et al. 2007) and merge-and-shrinks (Helmert, Haslum, and Hoffmann 2007) are represented explicitly by their induced transition systems, while structural patterns (Katz and Domshlak 2008b; 2009) correspond to implicitly, compactly represented abstractions.

One key feature of abstraction heuristics is that typically there is a great degree of flexibility in abstraction selection. This flexibility is a mixed blessing as the choice of abstraction may dramatically affect the quality of the heuristic estimate, while homing in on a better/best choice is not easy. A closer look at some successful approaches to both (explicit) pattern database abstractions and (implicit) structural pattern abstractions reveals some commonality in their strategies to resolve that choice dilemma. When a set of pattern databases is selected, the farther state variables are from the goal-mentioned variables $V(G)$ in the causal graph, the more they are likely to be abstracted away (ignored) altogether. The picture with the fork-decomposition structural patterns is very much similar. While there is not much room for flexibility in selecting such structural patterns, the size of the patterns' set, and thus the quality of the resulting heuristic, depend crucially on the number of goal-mentioned variables.

This dependence of some abstraction heuristics on the size of $V(G)$ is quite annoying as any SAS$^+$ planning problem can easily be reformulated to contain just a single goal-mentioned variable. However, things that were goals before the reformulation do not really cease to be goals, but only become *implicit goals*. In fact, this is just one type of possible implicit goals; in particular, *any landmark is such an implicit goal by its very definition.*

Landmark-based Problem Reformulation

Given the aforementioned dependence of some abstraction heuristics on the richness of the explicit goals, as well as the fact that many *de facto* goals are not explicitly given in the description of the problem, it is only natural to explore the possibility of converting some (discoverable) implicit goals to explicit goals. Probably the most direct way to achieve that is via a specific notion of one-sided equivalence between planning problems which we call *surrogate*. For a planning task Π, let Φ_Π denote the set of all optimal plans for Π. Given two planning tasks Π and Π', we say that Π' is a *surrogate* of Π if

(i) $\Phi_{\Pi'} = \emptyset$ iff $\Phi_\Pi = \emptyset$,

(ii) there exists a mapping $f : \Phi_{\Pi'} \rightarrow \Phi_\Pi$ such that, for any $\rho' \in \Phi_{\Pi'}$, $f(\rho')$ can be computed in time polynomial in $||\Pi||$, $|\rho'|$, and $|f(\rho')|$.

Note that, if Π' is a surrogate of Π, then instead of optimally solving Π, one can optimally solve Π', and then reconstruct an optimal plan for Π from the obtained plan for Π'. This is precisely what we exploit here using what we call *direct landmark-based surrogates*. Given a planning task $\Pi = \langle V, A, I, G, \mathcal{C}\rangle$, and a set of initially unachieved (that is, not true at time 0) disjunctive landmarks $L \subseteq 2^F$ of Π, the direct landmark-based surrogate $\Pi^L = \langle V^L, A^L, I^L, G^L, \mathcal{C}^L\rangle$ of Π is constructively defined as follows.

- For each landmark $\phi \in L$, we introduce a new variable v_ϕ with $dom(v_\phi) = \{0, 1\}$, and set $V^L = V \cup \{v_\phi \mid \phi \in L\}$.
- The initial state and goals are set to $I^L = I \cup \{v_\phi{=}0 \mid \phi \in L\}$, and $G^L = G \cup \{v_\phi{=}1 \mid \phi \in L\}$.
- For each action $a \in A$, we introduce an action

  $$a^L = \langle \mathsf{pre}(a),\ \mathsf{eff}(a) \cup \{v_\phi{=}1 \mid \mathsf{eff}(a)[v] \in \phi \in L\}\rangle,$$

  that is, for each landmark $\phi \in L$ that a can achieve, a^L will assign the corresponding auxiliary variable v_ϕ to its goal value. Given that, we set $A^L = \{a^L \mid a \in A\}$.

Proposition 1 *Given a planning task Π and a set of initially unachieved disjunctive landmarks L of Π, the direct landmark-based surrogate Π^L of Π is a surrogate of the latter, and can be constructed from Π and L in polynomial time.*

While the resemblance between the respective components of Π and Π^L is high, fork decomposition of Π^L both enriches the patterns that would be present in the fork decomposition of Π (by, e.g., adding more children to forks' roots), and induces patterns that would not be present in the fork decomposition of Π at all. In general, the richer is the fork-decomposition in terms of the number of patterns and the comprehensiveness of each pattern, the higher the estimate we can possibly obtain from that decomposition. However, "possibly obtain" and "obtain" are not necessarily the same thing, and a lot depends on the specific action cost partitioning between the patterns of the additive ensemble comprising $h^{\mathcal{F}}$ (Katz and Domshlak 2008a). The choice of action cost partitioning can vary from optimal (in terms of maximizing the estimate) to almost arbitrarily bad. The good news is that, under *optimal action cost partitioning* (achievable in polynomial time; see Katz and Domshlak 2008a), the dominance relation $h^{\mathcal{F}}(I^L) \geq h^{\mathcal{F}}(I)$ always holds. In fact, a stronger claim holds.

Proposition 2 *Given a planning task $\Pi = \langle V, A, I, G, \mathcal{C}\rangle$, and a direct landmark-based surrogate Π^L of Π, for any state s of Π and any state s' of Π^L such that $s'[V] = s$, under the optimal action cost partitioning we have $h^{\mathcal{F}}(s') \geq h^{\mathcal{F}}(s)$.*

While Proposition 2 seems to provide a clear-cut vote for the attractiveness of switching the search from Π to Π^L, in practice the picture is more complicated. The procedure of Katz and Domshlak (2008a) for devising an optimal action cost partition is polynomial-time, yet it is based on solving large linear programs, and thus takes too much time to be computed in practice at every search node. As the first sanity check for the practical usefulness of switching from Π to Π^L, we compared the initial-state estimates of a sub-optimal (yet cheep to compute) fork-decomposition heuristic for problems from a wide sample of IPC domains, as well as for their respective direct landmark surrogates.

- We have focused on the fork-decomposition heuristic $h^{\mathcal{F}}$, corresponding to the ensemble of all (not obviously redundant) fork subgraphs of the causal graph, with the domains of the roots being abstracted using the "leave-one-value-out" binary-valued domain decompositions; the action cost partitioning was set to "uniform" (Katz and Domshlak 2008b). On the domains considered in our evaluation this heuristic has exhibited the best performance among the family of fork-decomposition heuristics recently evaluated by Katz and Domshlak (2009).
- The landmarks were discovered using LAMA's landmark discovery procedure (Richter, Helmert, and Westphal 2008). The construction of the surrogate task Π^L from the original task Π was done as a part of the preprocessing.

Table 1 summarizes the effectiveness of switching to the surrogate problems in terms of the quality of the initial state estimation with the above setting of $h^{\mathcal{F}}$. These results appear to be very promising. Except for the Blocksworld domain where estimates on Π^L on average got slightly worse, in all other domains the estimates improved (or remained unchanged), with the most substantial average improvement of $\approx 120\%$ in Freecell, Grid, and Miconic. Given that LAMA's landmark discovery procedure typically takes very low time, these results suggest that the basic idea of incorporating landmark information into the process of problem abstraction is valuable. Next, however, we show that there is still some work to be done to make the basic idea "bulletproof".

domain	$\frac{h^{\mathcal{F}}(I)}{h^*(I)}$	$\frac{h^{\mathcal{F}}(I^L)}{h^*(I^L)}$
airport-ipc4 (20)	0.627	0.969
blocks-ipc2 (30)	0.477	0.432
depots-ipc3 (7)	0.419	0.508
driverlog-ipc3 (14)	0.632	0.708
freecell-ipc3 (7)	0.295	0.636
grid-ipc1 (2)	0.258	0.571
gripper-ipc1 (11)	0.375	0.546
logistics-ipc1 (6)	0.854	0.931
logistics-ipc2 (22)	0.998	0.994
miconic-strips-ipc2 (140)	0.434	0.964
mprime-ipc1 (25)	0.489	0.625
mystery-ipc1 (22)	0.621	0.713
openstacks-ipc5 (7)	0.611	0.829
pathways-ipc5 (5)	0.187	0.187
pipesworld-notankage-ipc4 (22)	0.302	0.359
pipesworld-tankage-ipc4 (14)	0.218	0.293
psr-small-ipc4 (50)	0.169	0.178
rovers-ipc5 (7)	0.529	0.653
satellite-ipc4 (9)	0.541	0.844
tpp-ipc5 (6)	0.838	0.836
trucks-ipc5 (9)	0.385	0.608
zenotravel-ipc3 (12)	0.694	0.809

Table 1: Average ratios between the initial state estimates of $h^{\mathcal{F}}$ on the original and landmarks-enhanced problems, and the optimal solution length. The first column captures the domains, with the number of problem instances taken into consideration in parentheses.

Estimating in Π^L, searching in Π

The two left-most groups of columns in Table 2, notably (1) $A^*/h^{\mathcal{F}}$ on Π^L and (2) $A^*/h^{\mathcal{F}}$ on Π, depict the performance of A^* with our setting of $h^{\mathcal{F}}$ on a sample of domains, with and without enriching fork-decomposition with landmarks, respectively. Columns *task* and h^* capture the identity of the problem instances, and their respective optimal solution cost; the performance is measured in terms of time (in seconds) and the number of expanded nodes. The implementation was based on the A^* algorithm with full duplicate elimination of the Fast Downward planner (Helmert 2006), and the structural-pattern database version of the $h^{\mathcal{F}}$ heuristic (Katz and Domshlak 2009). All the experiments were run on a 3GHz Intel E8400 CPU; the time and memory limits were set to 30 minutes and 1.5 GB, respectively.

On Satellite, Trucks, and Freecell, switching to Π^L substantially reduced the number of expanded nodes, and in the former two domains, allowed solving instances not solvable by A^* with $h^{\mathcal{F}}$ on the original problem formulations. With Blocksworld the picture was exactly the other way around, with the difference in the number of expanded nodes reaching almost three orders of magnitude in favor of not switching to Π^L. Given analysis of the initial state estimates in Table 1, such a qualitative outcome is not that surprising for Blocksworld. The picture, however, is more interesting in case of, e.g., Gripper and Driverlog domains. While the initial state estimates of $h^{\mathcal{F}}$ on these tasks surrogates are higher than on the original problem formulations,

		$A^*/h^{\mathcal{F}}$ on Π			$A^*/h^{\mathcal{F}}$ on Π^L			LM-A^* on Π; $h^{\mathcal{F}}$ on Π^L		
task	h^*	$h^{\mathcal{F}}(I)$	nodes	time	$h^{\mathcal{F}}(I^L)$	nodes	time	$h^{\mathcal{F}}(I^L)$	nodes	time
satellite-ipc4										
01	9	6	24	0.01	8	10	0.00	8	10	0.00
02	13	10	86	0.00	12	15	0.01	12	14	0.00
03	11	5	2249	0.11	9	1035	0.10	9	494	0.11
04	17	11	7510	0.56	15	4521	0.92	15	1046	0.48
05	15	7	279569	61.87	11	74442	17.37	11	12013	7.86
06	20	10	1496577	113.30	16	314788	65.61	16	40559	31.58
07	21				19	1593592	1120.68	19	236970	416.34
trucks-ipc5										
01	13	5	1691	0.06	8	450	0.04	8	262	0.05
02	17	7	9624	0.33	10	3126	0.28	10	1711	0.27
03	20	8	80693	4.24	12	13556	1.84	12	6465	1.74
04	23	8	1764624	64.95	12	991005	85.47	12	482690	171.31
05	25	9	12656561	662.25	15	4566431	533.04	15	1440802	1000.69
07	23	10	2135888	138.55	15	241608	45.98	15	84771	32.25
08	25				18	607186	215.22	18	97019	63.00
freecell-ipc3										
01	8	5	235	0.17	6	154	0.22	6	186	0.25
02	14	5	31050	2.98	9	6471	1.59	9	6471	3.01
03	18	6	197647	20.66	12	43476	12.07	12	39388	26.64
04	26	6	998395	87.19	16	186983	47.17	16	166323	120.90
05	30	5	6536337	630.56	17	735724	189.95	17	580174	443.27
blocks-ipc2										
04-0	6	6	15	0.00	6	9	0.00	6	15	0.01
04-1	10	4	13	0.01	4	31	0.02	4	17	0.02
04-2	6	5	8	0.00	4	11	0.01	4	11	0.01
05-0	12	5	35	0.01	5	187	0.03	5	99	0.03
05-1	10	6	42	0.00	4	124	0.02	4	79	0.02
05-2	16	7	151	0.00	6	680	0.05	6	264	0.05
06-0	12	9	33	0.01	7	339	0.06	7	105	0.05
06-1	10	9	42	0.01	7	212	0.05	7	104	0.04
06-2	20	8	932	0.01	7	5832	0.38	7	2097	0.33
07-0	20	10	265	0.01	8	16238	1.74	8	3317	0.80
07-1	22	9	6908	0.10	7	76747	5.83	7	23206	4.05
07-2	20	9	1552	0.03	8	37428	3.46	8	10467	1.98
08-0	18	11	1572	0.03	8	88029	12.99	8	18874	5.01
08-1	20	9	10470	0.18	8	294552	36.54	8	78864	18.96
08-2	16	12	532	0.02	9	24379	4.19	9	6233	1.74
09-0	30	12	137317	3.11	10	8778603	1051.08			
09-1	28	12	3530	0.10	9	2503352	390.23	9	364004	147.01
09-2	26	14	6328	0.17	11	2247311	358.67	11	203647	86.03
10-0	34	16	1548503	37.65						
10-1	32	16	615663	15.85						
10-2	34	16	1528645	37.28						
gripper-ipc1										
01	11	5	214	0.01	7	473	0.02	7	115	0.01
02	17	7	1768	0.04	10	10181	0.35	10	1403	0.12
03	23	9	11626	0.23	13	153356	6.92	13	10723	1.22
04	29	11	68380	1.65	16	2027535	119.67	16	66571	10.78
05	35	13	376510	11.04	19	24803228	1773.18	19	373331	77.19
06	41	15	1982032	75.66				22	1976923	536.39
07	47	17	10091986	465.70						
driverlog-ipc3										
01	7	3	49	0.01	5	17	0.01	5	12	0.00
02	19	12	15713	0.54	12	30831	1.99	12	15567	2.62
03	12	8	164	0.01	8	267	0.04	8	164	0.04
04	16	11	6161	0.55	10	10652	1.52	10	7367	2.07
05	18	12	13640	1.36	13	21202	4.50	13	8216	3.07
06	11	8	608	0.13	8	448	0.15	8	378	0.19
07	13	11	862	0.20	11	996	0.40	11	695	0.40
08	22	12	666282	97.11	13	1119854	334.70			
09	22	12	150237	19.19	14	194861	49.61	14	66879	32.51
10	17	12	4260	0.52	14	6369	3.06	14	2601	1.62
11	19	11	43395	6.97	14	29944	13.97	14	6174	3.89
13	26	15	1303099	473.73				18	233560	273.41

Table 2: A closer look on the performance of A^* with $h^{\mathcal{F}}$ under uniform action cost partitioning with ($A^*/h^{\mathcal{F}}$ on Π^L) and without ($A^*/h^{\mathcal{F}}$ on Π) enriching fork-decomposition with landmarks, as well as of LM-A^* with $h^{\mathcal{F}}$ where the search is done on the original problem Π, while the heuristic $h^{\mathcal{F}}$ is computed for states in Π^L.

the overall performance of A^* with $h^{\mathcal{F}}$ on the original problems appears to be at least as good (and in terms of the number of problems solved, much better than) on the surrogates Π^L.

While further investigation of this phenomenon is required, one of the potentially critical causes for that anomaly is the increase in the size of the search space from Π to Π^L. If a problem Π comes with a set of n landmarks L, then the number of reachable states in Π^L can be up to 2^n times larger than in Π because every state of Π is now considered in different contexts of achievement of different subsets of landmarks. To eliminate this cause to a large degree we have evaluated a scheme in which the heuristic estimates come from the surrogate problem while the (forward state-space) search is per-

formed on the original problem using the recent *LM-A** search algorithm of Karpas and Domshlak (2009). In general, this scheme works as follows.

1. Starting the initial state I, each evaluated state s of Π is first associated with a subset of landmarks $L_s \subseteq L$ that must be achieved on the way to the goal *from* s. Determining the landmark set L_s is done using the techniques developed by Richter et al. (2008) and Karpas and Domshlak (2009).

2. Given L_s, the state s is mapped into the state s' of Π^L where $s'[V] = s$, and, for each $\phi \in L$, if $\phi \in L_s$, then $s'[v_\phi] = 0$, otherwise, $s'[v_\phi] = 1$.

3. The heuristic estimate for s is set to $h^{\mathcal{F}}(s')$. The latter is no longer a state-dependent, but path, or even multi-path dependent estimate, and thus the machinery of *LM-A** is required. Importantly, however, this estimate is admissible, and thus *LM-A** guarantees to find only optimal solutions.

The third group of columns in Table 2 (*LM-A** on Π; $h^{\mathcal{F}}$ on Π^L) depict the results obtained with that scheme. In general, the reduction in the number of state expansions comparatively to running A^* directly on the surrogates Π^L has been consistently substantial. Note that, despite this improvement of pruning, some tasks solved with A^* on Π^L have not been solved using *LM-A** (e.g., task 8 in Driverlog), and this because of a generally higher search-node processing time of *LM-A**. However, the opposite has been observed as well (e.g., task 13 of Drivelog, and task 6 of Gripper), with the major difference being observed in the Miconic domain: while A^* with $h^{\mathcal{F}}$ solved 51 and 52 tasks from this domain on their original and surrogate representations, respectively, using the *LM-A** scheme with the same sets of landmarks allowed for solving 108 tasks in Miconic.

This, of course, is not the end of the story for landmarks and fork decomposition. As it is exemplified by the still not good results for the Blocksworld domain, the original growth of the search space was not the only source of troubles. We believe that another critical pitfall is the interplay between our ad hoc action cost partitioning among the patterns and the typical nature of landmarks. A landmark ϕ should take place only at *some* time point along the plan. Very roughly, once ϕ is achieved, the portions of the action costs "assigned to the achievement of ϕ" are getting lost, resulting in erosion of the heuristic values. In summary, devising better and yet fast, ad hoc action cost partitioning in face of landmark-oriented patterns is clearly desirable.

Future Work

We have outlined a general idea of exploiting landmark information for enhancing abstraction based heuristics, and proposed a concrete realization of this direction suitable for structural-pattern abstractions.

Many things are yet to be improved and extended even for the closely considered here case of structural-pattern abstractions. In addition, while the same landmark compilation technique can in principle be used in conjunction with other abstraction heuristics such as PDBs and merge-and-shrink, this will probably not be a good idea (and our selective experiments testified for that). The reason for that is not surprising: the particular problem reformulation induced by the direct landmark-based surrogates increases the dimensionality of the problem, up to possibly a linear factor. While structural-pattern abstractions are not sensitive to that matter, this is substantially less so for PDB and merge-and-shrink abstractions. Hence, some alternative ways of incorporating landmark information into abstractions should be looked for as well.

References

Bäckström, C., and Nebel, B. 1995. Complexity results for SAS$^+$ planning. *Comp. Intell.* 11(4):625–655.

Edelkamp, S. 2001. Planning with pattern databases. In *ECP*, 13–24.

Haslum, P., and Geffner, H. 2000. Admissible heuristics for optimal planning. In *ICAPS*, 140–149.

Haslum, P.; Botea, A.; Helmert, M.; Bonet, B.; and Koenig, S. 2007. Domain-independent construction of pattern database heuristics for cost-optimal planning. In *AAAI*, 1007–1012.

Helmert, M., and Domshlak, C. 2009. Landmarks, critical paths and abstractions: What's the difference anyway? In *ICAPS*.

Helmert, M.; Haslum, P.; and Hoffmann, J. 2007. Flexible abstraction heuristics for optimal sequential planning. In *ICAPS*, 176–183.

Helmert, M. 2006. The Fast Downward planning system. *JAIR* 26:191–246.

Hoffmann, J.; Porteous, J.; and Sebastia, L. 2004. Ordered landmarks in planning. *JAIR* 22:215–278.

Karpas, E., and Domshlak, C. 2009. Cost-optimal planning with landmarks. In *IJCAI*.

Katz, M., and Domshlak, C. 2008a. Optimal additive composition of abstraction-based admissible heuristics. In *ICAPS*, 174–181.

Katz, M., and Domshlak, C. 2008b. Structural patterns heuristics via fork decomposition. In *ICAPS*, 182–189.

Katz, M., and Domshlak, C. 2009. Structural-pattern databases. In *ICAPS*.

Porteous, J.; Sebastia, L.; and Hoffmann, J. 2001. On the extraction, ordering, and usage of landmarks in planning. In *ECP*, 37–48.

Richter, S.; Helmert, M.; and Westphal, M. 2008. Landmarks revisited. In *AAAI*, 975–982.