Bridging the Gap Between Refinement and Heuristics in Abstraction

Christer Bäckström and **Peter Jonsson**
Department of Computer Science, Linköping University
SE-581 83 Linköping, Sweden
christer.backstrom@liu.se peter.jonsson@liu.se

Abstract

There are two major uses of abstraction in planning and search: *refinement* (where abstract solutions are extended into concrete solutions) and *heuristics* (where abstract solutions are used to compute heuristics for the original search space). These two approaches are usually viewed as unrelated in the literature. It is reasonable to believe, though, that they are related, since they are both intrinsically based on the structure of abstract search spaces. We take the first steps towards formally investigating their relationships, employing our recently introduced framework for analysing and comparing abstraction methods. By adding some mechanisms for expressing metric properties, we can capture concepts like *admissibility* and *consistency* of heuristics. We present an extensive study of how such metric properties relate to the properties in the original framework, revealing a number of connections between the refinement and heuristic approaches. This also provides new insights into, for example, Valtorta's theorem and spurious states.

1 Introduction

One of the most widespread and important forms of abstraction in search and planning is state abstraction. In very general terms, this means that one forms an abstract state space from the original state space. The purpose of the abstract space is that it may help us to solve the original problem faster. Two dominating methods exist for exploiting state abstraction for this purpose: refinement and heuristics.

Refinement planning was pioneered in the ABSTRIPS planner [Sacerdoti, 1974], but the idea was used already in GPS [Newell *et al.*, 1959]. Refinement first finds a plan in the abstract version of the problem instance and then uses this plan as a skeleton for a plan for the original instance. If one is lucky, then it is sufficient to add more actions between the ones in this skeleton plan to obtain a valid plan. Otherwise, one has to backtrack and find a new skeleton plan. The abstraction heuristic method instead performs heuristic search in the original search space, using the abstract space to compute the heuristic. The heuristic search approach has proven very successful both in classical planning and in search, and a number of interesting and well-performing heuristics have been invented (see Helmert and Domshlak [2009] for a comprehensive survey and comparison).

Refinement used to be the dominating abstraction method in planning, but has largely been replaced with abstraction heuristics, although there are signs of a potential renaissance for refinements [Gregory *et al.*, 2011; Seipp and Helmert, 2013]. Refinement has continued to be used, though, in other areas such as path planning [Sturtevant and Buro, 2005] and model checking [Clarke *et al.*, 2003]. The refinement and heuristic approaches are usually considered as quite different and unrelated. That is a very superficial analysis, though: both approaches are firmly based on properties of the solutions in the abstract space, so some connections are bound to exist. Yet, the literature is almost void of attempts to investigate these connections. A notable exception is Helmert [2006] who made a pragmatic attempt at combining refinement and heuristics in his FD planner. A formal example is the **DPP** criterion [Zilles and Holte, 2010], intended to avoid certain bad types of abstractions in heuristic search, but also related to the backtracking issue.

The purpose of this paper is twofold. One is to extend our own previous abstract framework for modelling abstractions [Bäckström and Jonsson, 2012a; 2012b] by adding metric properties in addition to the previous qualitative ones. The other purpose is to use this extended framework to provide the first formal analysis on an abstract method-independent level of how refinement and heuristics are related. This paper should not be viewed as a solitary one, but be read in the context of our previous publication on this framework.

The rest of the paper is structured as follows. Sections 2 and 3 recapitulate the abstraction framework that we use, and Section 4 discusses the three different refinement concepts previously studied within this framework. Our main results appear in Section 5, where we add new metric properties to the framework in order to express concepts like admissibility and consistency of heuristics. We further make an extensive investigation of how these metric properties relate to the previously proposed properties. An immediate consequence of this is that we identify several connections between refinement and heuristics. Section 6 gives further examples of using the extended framework to study various concepts in a different and more revealing way, eg. Valtorta's theorem and the **DPP** criterion. The paper ends with a discussion section.

2 STGs and STG Transformations

We here briefly present the framework for studying abstractions and refer to our previous papers [Bäckström and Jonsson, 2012a; 2012b] for further details and explanations.

Let X be a set. Then $|X|$ denotes its cardinality. A *partition* of X is a set P of non-empty subsets of X s.t. (1) $\cup_{p\in P} p = X$ and (2) for all $p, q \in P$, if $p \neq q$, then $p \cap q = \varnothing$. Let $f : X \to Y$ be a function, then $Rng(f) = \{f(x) \mid x \in X\}$ is the *range* of f. When the elements of Y are sets we define $f(Z) = \cup_{x\in Z} f(x)$ for all $Z \subseteq X$.

Definition 1. A *state transition graph (STG)* over a set L of labels is a tuple $\mathbb{G} = \langle S, E\rangle$ where S is a set of vertices and $E \subseteq S \times S \times L$ is a set of labelled arcs. Also define $L(\mathbb{G}) = \{\ell \mid \langle s, t, \ell\rangle \in E\}$. A sequence $s_0, s_1, \ldots, s_k$ of states in S is a *path* in $\mathbb{G}$ if either (1) $k = 0$ or (2) $\langle s_{i-1}, s_i, \ell_i\rangle \in E$ for $1 \leq i \leq k$ and some labels $\ell_1, \ldots, \ell_k$.

The set S is called a *state space* and its members *states*. We allow multiple arcs between two states if they differ in direction or labels. Labels provide a means to identify sets of arcs, eg. the arcs induced by a particular action, but they are not of much relevance in this paper. The STG collapses to an ordinary directed graph if all arcs have the same label.

Definition 2. Let $\mathbb{G}_1 = \langle S_1, E_1\rangle$ and $\mathbb{G}_2 = \langle S_2, E_2\rangle$ be two STGs. A total function $f : S_1 \to 2^{S_2}$ is a *transformation function* from $\mathbb{G}_1$ to $\mathbb{G}_2$ if $Rng(f)$ is a partition of S_2. The corresponding *reverse transformation function* $\overline{f} : S_2 \to 2^{S_1}$ is defined as $\overline{f}(t) = \{s \in S_1 \mid t \in f(s)\}$. A *label relation* from $\mathbb{G}_1$ to $\mathbb{G}_2$ is a binary relation $R \subseteq L(\mathbb{G}_1) \times L(\mathbb{G}_2)$. The *reverse label relation* $\overline{R} \subseteq L(\mathbb{G}_2) \times L(\mathbb{G}_1)$ is defined as $\overline{R}(\ell_2, \ell_1)$ iff $R(\ell_1, \ell_2)$. A *transformation* from $\mathbb{G}_1$ to $\mathbb{G}_2$ is a pair $\tau = \langle f, R\rangle$ where f is a transformation function from $\mathbb{G}_1$ to $\mathbb{G}_2$ and R is a label relation from $\mathbb{G}_1$ to $\mathbb{G}_2$.

Intuitively, f specifies how τ maps states in $\mathbb{G}_1$ to sets of states in $\mathbb{G}_2$, while R relates subsets of E_1 with subsets of E_2. Transformation functions are extended to sequences of states such that $f(s_1, \ldots, s_k) = f(s_1), \ldots, f(s_k)$. Furthermore, if $\tau = \langle f, R\rangle$ is a transformation, then: (1) $s \in \overline{f}(t)$ if and only if $t \in f(s)$, (2) $\overline{f}$ is a transformation function from $\mathbb{G}_2$ to $\mathbb{G}_1$ and (3) $\langle \overline{f}, \overline{R}\rangle$ is a transformation from $\mathbb{G}_2$ to $\mathbb{G}_1$. As a convention, we will often not specify the STGs in definitions and theorems, tacitly assuming transformations to be from an STG $\mathbb{G}_1 = \langle S_1, E_1\rangle$ to an STG $\mathbb{G}_2 = \langle S_2, E_2\rangle$ unless otherwise specified. We also sometimes refer to $\mathbb{G}_1$ as the *ground* graph and $\mathbb{G}_2$ as the *abstract* graph.

3 Method Properties

We have earlier defined a number of properties that transformations can have, dividing these into *method propeties* and *instance properties* [Bäckström and Jonsson, 2012a]. In brief, method properties are inherent for a particular transformation method and the actual STGs do not matter, while instance properties may hold only for particular pairs of STGs.

Definition 3. A transformation $\tau = \langle f, R\rangle$ can have the following *method properties*:

$\mathbf{M}_\uparrow$: $|f(s)| = 1$ for all $s \in S_1$.

$\mathbf{M}_\downarrow$: $|\overline{f}(s)| = 1$ for all $s \in S_2$.

$\mathbf{R}_\uparrow$: If $\langle s_1, t_1, \ell_1\rangle \in E_1$, then there is some $\langle s_2, t_2, \ell_2\rangle \in E_2$ such that $R(\ell_1, \ell_2)$.

$\mathbf{R}_\downarrow$: If $\langle s_2, t_2, \ell_2\rangle \in E_2$, then there is some $\langle s_1, t_1, \ell_1\rangle \in E_1$ such that $R(\ell_1, \ell_2)$.

$\mathbf{C}_\uparrow$: If $R(\ell_1, \ell_2)$ and $\langle s_1, t_1, \ell_1\rangle \in E_1$, then there is some $\langle s_2, t_2, \ell_2\rangle \in E_2$ such that $s_2 \in f(s_1)$ and $t_2 \in f(t_1)$.

$\mathbf{C}_\downarrow$: If $R(\ell_1, \ell_2)$ and $\langle s_2, t_2, \ell_2\rangle \in E_2$, then there is some $\langle s_1, t_1, \ell_1\rangle \in E_1$ such that $s_1 \in \overline{f}(s_2)$ and $t_1 \in \overline{f}(t_2)$.

We refer the reader to the original papers for further explanations of these properties. However, we provide a new alternative way to understand the properties, in terms of graph morphisms. We define the latter in the usual way for directed labelled graphs, ignoring the labels.

Definition 4. An $\mathbf{M}_\uparrow$ transformation function f from $\mathbb{G}_1$ to $\mathbb{G}_2$ is (1) a *homomorphism* if for all $s, t \in S_1$, $\langle s, t\rangle \in E_1$ implies $\langle f(s), f(t)\rangle \in E_2$; (2) a *strong homomorphism* if it is a homomorphism and for every $\langle s, t\rangle \in E_2$, there is some $\langle s', t'\rangle \in E_1$ s.t. $f(s') = s$ and $f(t') = t$; (3) an *embedding* if f is a bijection that is a homomorphism ; (4) a *retraction* if it is an embedding from $\mathbb{G}_2$ to $\mathbb{G}_1$.

These concepts have been extensively used as abstraction functions in search and planning. It turns out that the method properties suffice to capture and distinguish between these different concepts, which strongly indicates that the method properties are not arbitrary but express something essential.

Theorem 5. *If f is a transformation function then there is a label relation R s.t. the transformation $\tau = \langle f, R\rangle$ is:*

1) $\boldsymbol{M}_\uparrow\boldsymbol{R}_\uparrow\boldsymbol{C}_\uparrow$ if and only if f is a homomorphism.
2) $\boldsymbol{M}_\uparrow\boldsymbol{R}_\updownarrow\boldsymbol{C}_\updownarrow$ if and only if f is a strong homomorphism.
3) $\boldsymbol{M}_\updownarrow\boldsymbol{R}_\uparrow\boldsymbol{C}_\uparrow$ if and only if f is an embedding.
4) $\boldsymbol{M}_\updownarrow\boldsymbol{R}_\downarrow\boldsymbol{C}_\downarrow$ if and only if f is a retraction.

Proof sketch. (1-2) Follows from Theorem 9 in Bäckström and Jonsson [2012a] since part 1 of the proof corresponds to homomorphism and part 2 to the additional criterion for strong homomorphisms. (3) Immediate from 1 plus definitions. (4) Immediate from 3. □

We write $\mathbf{X} \Rightarrow \mathbf{Y}$ to denote that every transformation that has property $\mathbf{X}$ also must have property $\mathbf{Y}$, and we write $\mathbf{X} \not\Rightarrow \mathbf{Y}$ when this is not the case.

4 Path Refinement

It is often useful to consider an abstraction of a graph in terms of *soundness* and *completeness*. Loosely speaking, it is sound if every abstract path somehow correspond to a ground path and it is complete if every ground path somehow correspond to an abstract path. One way to formalize these concepts is to consider them in terms of *state refinement*. While it is common in planning to refine an abstract plan by using the corresponding ground actions as a skeleton plan, it is also possible to just use the ground states corresponding to the states along the abstract plan. In fact, this is the common way to do refinement in search, and Holte *et al.* [1996a] suggest that this is indeed a better way to do refinement.

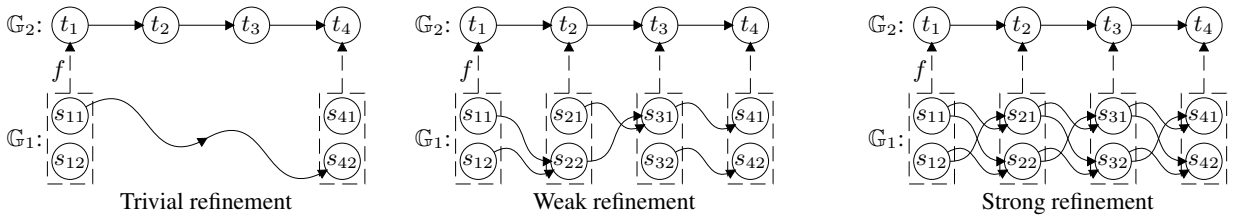

Figure 1: Downward path refinement.

We have previously defined three different types of state refinement, corresponding to different degrees of avoiding backtracking between levels [Bäckström and Jonsson, 2012a]. Very briefly, all three refinements avoid backtracking to the abstract level but correspond to different amounts of backtracking on the ground level.

Definition 6. Let f be a transformation function and let $\sigma = t_0, t_1, \ldots, t_k$ be an arbitrary path in $\mathbb{G}_2$. Then: 1) σ is *trivially downward state refinable* if there are two states $s_0 \in \overline{f}(t_0)$ and $s_\ell \in \overline{f}(t_k)$ s.t. there is a path in $\mathbb{G}_1$ from s_0 to s_ℓ. 2) σ is *weakly downward state refinable* if there is a sequence $s_0, s_1, \ldots, s_k$ of states in S_1 such that $s_i \in \overline{f}(t_i)$ for all i s.t. $0 \leq i \leq k$ and there is a path from s_{i-1} to s_i in $\mathbb{G}_1$ for all i ($1 \leq i \leq k$). 3) σ is *strongly downward state refinable* if for every i s.t. $1 \leq i \leq k$, there is a path from s_{i-1} to s_i in $\mathbb{G}_1$ for all $s_{i-1} \in \overline{f}(t_{i-1})$ and all $s_i \in \overline{f}(t_i)$.

These concepts are illustrated in Figure 1. All three cases consider the same abstraction and the same path $\sigma = t_1 t_2 t_3 t_4$ in the abstract graph, but different ground graphs. Curly arrows denote paths, i.e. they may consist of several arcs and pass through states not shown in the figure. Trivial refinement only requires that there is a path corresponding to σ in the ground graph. In this case it requires that there is a path from some state in $\overline{f}(t_1) = \{s_{11}, s_{12}\}$ to some state in $\overline{f}(t_4) = \{s_{41}, s_{42}\}$, which is satisfied by the path from s_{11} to s_{42}. Weak refinement additionally requires that we use all states along σ and also pass through some state in each of $\overline{f}(t_2)$ and $\overline{f}(t_3)$. This is satisfied, for instance, by the path $s_{11} s_{22} s_{31} s_{41}$. Finally, strong refinement requires that there is a path for any choice of states in $\overline{f}(t_1), \ldots, \overline{f}(t_4)$, which is satsified in the last example in the figure.

To continue, we must define reachability in graphs. Let $\mathbb{G} = \langle S, E \rangle$ be an STG. Then for all $s \in S$, the set $\mathcal{R}(s)$ of *reachable states* from s is defined as $\mathcal{R}(s) = \{t \in S \mid \text{ there is a path from } s \text{ to } t \text{ in } \mathbb{G}\}$. We extend this s.t. for all $T \subseteq S$, $\mathcal{R}(T) = \cup_{s \in T} \mathcal{R}(s)$. Using $\mathcal{R}_1(\cdot)$ for reachability in $\mathbb{G}_1$ and $\mathcal{R}_2(\cdot)$ for reachability in $\mathbb{G}_2$, instance properties are defined as follows.

Definition 7. A transformation function f can have the following *instance properties*:

$\mathbf{P_{k\downarrow}}$: For every path $t_0, \ldots, t_k$ in S_2, there are $s_0, \ldots, s_k \in S_1$ s.t. $s_i \in \overline{f}(t_i)$ for all i ($0 \leq i \leq k$) and $s_i \in \mathcal{R}_1(s_{i-1})$ for all i ($1 \leq i \leq k$).

$\mathbf{P_{k\uparrow}}$: For every path $s_0, \ldots, s_k$ in S_1, there are $t_0, \ldots, t_k \in S_2$ s.t. $t_i \in f(s_i)$ for all i ($0 \leq i \leq k$) and $t_i \in \mathcal{R}_2(t_{i-1})$ for all i ($1 \leq i \leq k$).

$\mathbf{P_{T\downarrow}}$: $\mathbf{P}_{1\downarrow}$ holds.

$\mathbf{P_{T\uparrow}}$: $\mathbf{P}_{1\uparrow}$ holds.

$\mathbf{P_{W\downarrow}}$: $\mathbf{P}_{k\downarrow}$ holds for all $k > 0$.

$\mathbf{P_{W\uparrow}}$: $\mathbf{P}_{k\uparrow}$ holds for all $k > 0$.

$\mathbf{P_{\downarrow}}$: If $t \in \mathcal{R}_2(f(s))$, then $\overline{f}(t) \cap \mathcal{R}_1(s) \neq \varnothing$.

$\mathbf{P_{\uparrow}}$: If $t \in \mathcal{R}_1(s)$, then $f(t) \cap \mathcal{R}_2(f(s)) \neq \varnothing$.

$\mathbf{P_{S\downarrow}}$: If $t \in \mathcal{R}_2(f(s))$, then $\overline{f}(t) \subseteq \mathcal{R}_1(s)$.

$\mathbf{P_{S\uparrow}}$: If $t \in \mathcal{R}_1(s)$, then $f(t) \subseteq \mathcal{R}_2(f(s))$.

The following relationships are known to hold.
$\mathbf{P_{S\downarrow}} \Rightarrow \mathbf{P_{\downarrow}} \Rightarrow \mathbf{P_{W\downarrow}} \Rightarrow \mathbf{P_{T\downarrow}}$ $\quad \mathbf{P_{T\downarrow}} \not\Rightarrow \mathbf{P_{W\downarrow}} \not\Rightarrow \mathbf{P_{\downarrow}} \not\Rightarrow \mathbf{P_{S\downarrow}}$
The instance properties capture refinement as follows.

Theorem 8. *(Bäckström and Jonsson, 2012a, Th. 15) For a transformation $\tau = \langle f, R \rangle$, every path in $\mathbb{G}_2$ is trivially/weakly/strongly downward state refinable iff τ is $\mathbf{P_{T\downarrow}}$/$\mathbf{P_{W\downarrow}}$/$\mathbf{P_{S\downarrow}}$.*

5 Metrics and Heuristics

In this section we will augment our framework with a few metric properties, in addition to the previous qualitative ones, and investigate how these properties relate to each other. In particular, we will analyse on an abstract method-independent level how refinement and heuristics relate to each other. The results we will prove in this section are summarized in Figure 2, where the arrows denote the $\Rightarrow$ and $\not\Rightarrow$ relationships between the properties (the figure contains the new properties that are yet to be defined).

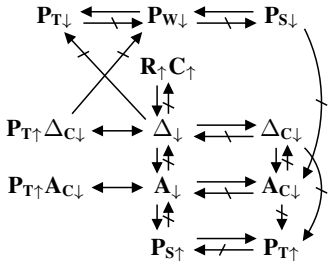

Figure 2: Relationships for $\mathbf{M}_{\uparrow}$ transformations.

5.1 Metric Properties

Heuristic search attempts to find an optimal solution faster than blind search by using a heuristic function that approximates the true cost to guide the search. It is desirable that

this function is admissible, i.e. that it never overestimates the true cost. For instance, the A* algorithm is optimal under this condition [Dechter and Pearl, 1985]. It is often useful to view both the original and abstract search spaces as graphs and to define the abstraction such that the length of paths in the abstract graph is an admissible heuristic for the length or cost of the corresponding paths in the ground graph [Holte *et al.*, 1996a]. That is, abstraction is viewed as a graph transformation. It is also common to preprocess the abstract graph and store the path lengths in a pattern database [Culberson and Schaeffer, 1998], or to compute a heuristic from several such databases [Haslum *et al.*, 2007].

Define $\mathbb{N}^\infty = \mathbb{N} \cup \{\infty\}$ and extend $=$, $<$ and $+$ to $\mathbb{N}^\infty$ in the obvious way. Let $\mathbb{G} = \langle S, E\rangle$ be an STG. A *cost function* for $\mathbb{G}$ is a function $c : S^2 \to \mathbb{N}^\infty$, with the restriction that for all $s, t \in S$, $c(s,t) = \infty$ if and only if there is no path from s to t in $\mathbb{G}$. A *heuristic function* h for a cost function c is itself a cost function. As usual, h is *admissible* for c if $0 \leq h(s,t) \leq c(s,t)$, for all $s, t \in S$ and *consistent* for c if $h(s,t) \leq c(s,u) + h(u,t)$ for all $s, t, u \in S$. While s and t are sometimes assumed to be the initial and goal states, respectively, our definition is more general and is also common, cf. Yang *et al.* [2008]. It is better suited for our purpose of comparing refinement with heuristics, and it is a reasonable assumption for domain-independent heuristics. While truly arbitrary cost functions are sometimes considered, this is usually not the case in planning and many other contexts. We follow common practice and define cost functions as follows. First define a function $w : E \to \mathbb{N}$ that assigns a *weight* to each arc in $\mathbb{G}$. Then extend the cost function to paths such that $c(s_0, \ldots, s_k) = \sum_{i=1}^{k} w(s_{i-1}, s_i)$. The cost $c(s,t)$ between two states is then defined as the minimum of $c(\sigma)$ over all paths σ from s to t, or ∞ when there is no path at all, ie. $c(s,t)$ is the cost of the cheapest path from s to t. We also implicitly define the specific weight function d which assigns weight 1 to all arcs, which is known as the *unit cost assumption*. When considering two STGs $\mathbb{G}_1$ and $\mathbb{G}_2$ simultaneously we index their corresponding c, d and w functions analogously, for instance, w_1 is the weight function for $\mathbb{G}_1$.

We extend transformations with metric information, writing $\tau = \langle f, R, w_1, w_2\rangle$ where $\langle f, R\rangle$ is a transformation from $\mathbb{G}_1$ to $\mathbb{G}_2$ and w_1 and w_2 are the weight functions for $\mathbb{G}_1$ and $\mathbb{G}_2$, respectively. The cost functions c_1 and c_2 are implicitly defined by w_1 and w_2. For instance, the common practice of using the path length in the abstract graph as a heuristic estimate for the path length in the ground graph correspdonds to an $\langle f, R, d_1, d_2\rangle$ transformation, while an $\langle f, R, w_1, d_2\rangle$ transformation estimates path costs in the ground graph with path lengths in the abstract graph. In order to take advantage of cost functions we will introduce some new metric properties. In this section, we only consider such properties for $\mathbf{M}_\uparrow$ transformation functions, since abstraction heuristics usually assume that f is an ordinary function.

Definition 9. An $\mathbf{M}_\uparrow$ transformation $\tau = \langle f, R, w_1, w_2\rangle$ can have the following *metric properties*:

$\mathbf{A}_\downarrow$: $c_2(f(s), f(t)) \leq c_1(s,t)$ for all $s, t \in S_1$.

$\mathbf{A_{C\downarrow}}$: $c_2(f(s), f(t)) \leq c_1(s,t)$ or $c_2(f(s), f(t)) = \infty$ for all $s, t \in S_1$.

$\Delta_\downarrow$: $c_2(f(s), f(t)) \leq c_1(s,u) + c_2(f(u), f(t))$ for all $s, t, u \in S_1$.

$\Delta_{\mathbf{C}\downarrow}$: $c_2(f(s), f(t)) \leq c_1(s,u) + c_2(f(u), f(t))$ or $c_2(f(s), f(t)) = \infty$ for all $s, t, u \in S_1$.

Properties $\mathbf{A}_\downarrow$ and $\Delta_\downarrow$ correspond to admissibility and consistency, respectively. Properties $\mathbf{A_{C\downarrow}}$ and $\Delta_{\mathbf{C}\downarrow}$ are conditional variants of $\mathbf{A}_\downarrow$ and $\Delta_\downarrow$ for transformations that are incomplete; they are only required to hold in the cases where there actually is a path in $\mathbb{G}_2$. More precisely, we have:

Theorem 10. *Let $\tau = \langle f, R, w_1, w_2\rangle$ be an $\mathbf{M}_\uparrow$ transformation. Then: 1) τ is $\mathbf{A}_\downarrow$ iff c_2 is an admissible heuristic for c_1. 2) τ is $\Delta_\downarrow$ iff c_2 is a consistent heuristic for c_1.*

Property $\mathbf{A}_\downarrow$ is a weaker criterion than $\Delta_\downarrow$, while $\mathbf{A_{C\downarrow}}$ and $\Delta_{\mathbf{C}\downarrow}$ are weaker variants of $\mathbf{A}_\downarrow$ and $\Delta_\downarrow$. The relationships between the metric properties can be summarized as follows.

Theorem 11. *Let τ be an $\mathbf{M}_\uparrow$ transformation. Then: 1) $\mathbf{A}_\downarrow \Rightarrow \mathbf{A_{C\downarrow}}$, 2) $\Delta_\downarrow \Rightarrow \Delta_{\mathbf{C}\downarrow}$, 3) $\mathbf{A_{C\downarrow}} \not\Rightarrow \mathbf{A}_\downarrow$, 4) $\Delta_{\mathbf{C}\downarrow} \not\Rightarrow \Delta_\downarrow$, 5) $\Delta_\downarrow \Rightarrow \mathbf{A}_\downarrow$, 6) $\Delta_{\mathbf{C}\downarrow} \Rightarrow \mathbf{A_{C\downarrow}}$, 7) $\mathbf{A}_\downarrow \not\Rightarrow \Delta_\downarrow$, 8) $\mathbf{A_{C\downarrow}} \not\Rightarrow \Delta_{\mathbf{C}\downarrow}$.*

The results in the remainder of this section will fill in the rest of the arrows in Figure 2.

5.2 Metric Properties and Upward Refinement

The following theorem formalizes that admissibility implies completeness, but the opposite is false; not even the strongest form of completeness, $\mathbf{P_{S\uparrow}}$, guarantees admissibility.

Theorem 12. *Let τ be an $\mathbf{M}_\uparrow$ transformation. Then:*
1) $\mathbf{A}_\downarrow \Rightarrow \mathbf{P_{S\uparrow}}$ 2) $\Delta_\downarrow \Rightarrow \mathbf{P_{S\uparrow}}$ 3) $\mathbf{P_{S\uparrow}} \not\Rightarrow \mathbf{A}_\downarrow$
4) $\mathbf{P_{S\uparrow}} \not\Rightarrow \Delta_\downarrow$ 5) $\mathbf{A_{C\downarrow}} \not\Rightarrow \mathbf{P_{T\uparrow}}$ 6) $\Delta_{\mathbf{C}\downarrow} \not\Rightarrow \mathbf{P_{T\uparrow}}$

Proof. Let $\tau = \langle f, R, w_1, w_2\rangle$ be an $\mathbf{M}_\uparrow$ transformation.

1-2) Suppose τ is $\mathbf{A}_\downarrow$. Let $s, t \in S_1$ arbitrary such that $t \in \mathcal{R}_1(s)$. Then $c_1(s,t) < \infty$ and, thus, $c_2(f(s), f(t)) < \infty$, since τ is $\mathbf{A}_\downarrow$. Hence, $f(t) \subseteq \mathcal{R}_2(f(s))$ so τ is $\mathbf{P_{S\uparrow}}$ since s and t were chosen arbitrarily. (2) Follows since $\Delta_\downarrow \Rightarrow \mathbf{A}_\downarrow$.

3) Let $S_1 = S_2 = \{s_1, s_2, s_3\}$, $E_1 = \{\langle s_1, s_2, \ell\rangle, \langle s_1, s_3, \ell\rangle, \langle s_2, s_3, \ell\rangle\}$ and $E_2 = \{\langle s_1, s_2, \ell\rangle, \langle s_2, s_3, \ell\rangle\}$. Let f be the identity function, $R = \{\langle \ell, \ell\rangle\}$, $w_1 = d_1$ and $w_2 = d_2$. Then, τ is $\mathbf{P_{S\uparrow}}$. However, it is not $\mathbf{A}_\downarrow$ since $c_1(s_1, s_3) = d_1(s_1, s_3) = 1$ but $c_2(f(s_1), f(s_3)) = d_2(f(s_1), f(s_2)) + d_2(f(s_2), f(s_3)) = 2$. (4) Follows since $\Delta_\downarrow \Rightarrow \mathbf{A}_\downarrow$.

5,6) Let $E_1 \neq \varnothing$ and $E_2 = \varnothing$. Then τ is vacuously $\mathbf{A_{C\downarrow}}$ and $\Delta_{\mathbf{C}\downarrow}$ but not $\mathbf{P_{T\uparrow}}$. □

Figure 3 is an example of a $\mathbf{P_{S\uparrow}}$ transformation. However, there is a one-arc path s_{12}, s_{32} in $\mathbb{G}_1$ but the shortest path from $f(s_{12})$ to $f(s_{32})$ in $\mathbb{G}_2$ is of length 2. If we apply unit cost to both graphs, then the transformation cannot be $\mathbf{A}_\downarrow$.

Property $\mathbf{A}_\downarrow$ captures admissibility, which is complete, while $\mathbf{A_{C\downarrow}}$ is a conditional variant that does not require completeness but only that the abstraction does not overestimate the cost of a ground path whenever there is a corresponding abstract path. It is, however, sufficient to combine conditional admissibility with the weakest form of completeness to get full admissibility, as the following theorem demonstrates. The analogous case holds for consistency.

Theorem 13. *Let τ be an $\mathbf{M}_\uparrow$ transformation. Then:*
1) $\mathbf{P_{T\uparrow}}\mathbf{A_{C\downarrow}} \Leftrightarrow \mathbf{A}_\downarrow$ 2) $\mathbf{P_{T\uparrow}}\Delta_{\mathbf{C}\downarrow} \Leftrightarrow \Delta_\downarrow$

5.3 Metric Properties and Downward Refinement

Admissibility enforces completeness but not soundness; that is, there can be an abstract path with no corresponding ground path. In fact, admissibility and downward refinement are largely orthogonal and incomparable concepts.

Theorem 14. *Let τ be an $\mathbf{M}_\uparrow$ transformation. Then:*
1) $\mathbf{P_{S\downarrow}} \not\Rightarrow \mathbf{A_{C\downarrow}}$ 2) $\mathbf{P_{S\downarrow}} \not\Rightarrow \Delta_{\mathbf{C}\downarrow}$ 3) $\mathbf{A}_\downarrow \not\Rightarrow \mathbf{P_{T\downarrow}}$
4) $\Delta_\downarrow \not\Rightarrow \mathbf{P_{T\downarrow}}$ 5) $\mathbf{P_{T\downarrow}A}_\downarrow \not\Rightarrow \mathbf{P_{W\downarrow}}$ 6) $\mathbf{P_{T\downarrow}}\Delta_\downarrow \not\Rightarrow \mathbf{P_{W\downarrow}}$

Proof. 1-2) Let $S_1 = S_2 = \{s_1, s_2, s_3\}$, $E_1 = \{\langle s_1, s_2, \ell\rangle, \langle s_1, s_3, \ell\rangle, \langle s_2, s_3, \ell\rangle\}$ and $E_2 = \{\langle s_1, s_2, \ell\rangle, \langle s_2, s_3, \ell\rangle\}$. Also let f be the identity function, $R = \{\langle \ell, \ell\rangle\}$ and $\tau = \langle f, R, d_1, d_2\rangle$. Then τ is $\mathbf{P_{S\downarrow}}$. However, $c_1(s_1, s_3) = d_1(s_1, s_3) = 1$ but $c_2(f(s_1), f(s_3)) = d_2(f(s_1), f(s_2)) + d_2(f(s_2), f(s_3)) = 2$, so τ is not $\mathbf{A_{C\downarrow}}$. Hence, τ is not $\Delta_{\mathbf{C}\downarrow}$ since $\Delta_{\mathbf{C}\downarrow} \Rightarrow \mathbf{A_{C\downarrow}}$.

3-4) Let $S_1 = S_2 = \{s_1, s_2, s_3\}$, $E_1 = \varnothing$ and $E_2 = \{\langle s_1, s_2, \ell\rangle, \langle s_2, s_3, \ell\rangle\}$. Also let f be the identity function, $R = \{\langle \ell, \ell\rangle\}$ and $\tau = \langle f, R, d_1, d_2\rangle$. Then τ is $\mathbf{A}_\downarrow$ and $\Delta_\downarrow$ but it is not $\mathbf{P_{T\downarrow}}$ since $E_1 = \varnothing$.

5-6) Let $S_1 = \{s_{ij} \mid 1 \le i \le 4, 1 \le j \le 2\}$, $S_2 = \{s_1, s_2, s_3, s_4\}$, let $E_1 = \{\langle s_{11}, s_{21}, \ell\rangle, \langle s_{11}, s_{31}, \ell\rangle, \langle s_{31}, s_{41}, \ell\rangle, \langle s_{22}, s_{42}, \ell\rangle\}$ and let $E_2 = \{\langle s_1, s_2, \ell\rangle, \langle s_2, s_4, \ell\rangle, \langle s_1, s_3, \ell\rangle, \langle s_3, s_4, \ell\rangle\}$. Also let $f(s_{ij}) = s_i$ for all $s_{ij} \in S_1$, let $R = \{\langle \ell, \ell\rangle\}$ and let $\tau = \langle f, R, d_1, d_2\rangle$. Then τ is obviously $\mathbf{P_{T\downarrow}}$, $\mathbf{A}_\downarrow$ and $\Delta_\downarrow$. However, the path s_1, s_2, s_4 in $\mathbb{G}_2$ is not weakly refinable, so τ is not $\mathbf{P_{W\downarrow}}$. □

The major reason for these results is that refinement is defined without any metrics; even if we have a guarantee that an abstract plan can be refined into a ground plan, we have no guarantee that there is no shorter ground plan.

5.4 Relating Metric and Method Properties

Also method properties have metric connections.

Theorem 15. *Let $\tau = \langle f, R, w_1, w_2\rangle$ be an $\mathbf{M}_\uparrow\mathbf{R}_\uparrow\mathbf{C}_\uparrow$ transformation. Then: 1) If $w_2(f(s), f(t)) \le w_1(s, t)$ for all $\langle s, t\rangle \in E_1$ such that $\langle f(s), f(t)\rangle \in E_2$, then τ is $\mathbf{A}_\downarrow\Delta_\downarrow$. 2) Otherwise, τ need not be neither $\mathbf{A}_\downarrow$ nor $\Delta_\downarrow$.*

Proof. 1) It follows from Theorem 5 that f is a homomorphism since τ is $\mathbf{M}_\uparrow\mathbf{R}_\uparrow\mathbf{C}_\uparrow$. Hence, for every path σ in $\mathbb{G}_1$ also $f(\sigma)$ is a path in $\mathbb{G}_2$. Let $s, t \in S_1$ be arbitrary states. If there is no path from s to t in $\mathbb{G}_1$, then $c_1(s, t) = \infty$ so $c_2(f(s), f(t)) \le c_1(s, t)$ holds trivially. Otherwise, let $\sigma = s_0, \ldots, s_k$, where $s_0 = s$ and $s_k = t$, be the cheapest path from s to t. Then $c_1(s, t) = \sum_{i=1}^{k} w_1(s_{i-1}, s_i)$. Since also $f(\sigma)$ must be a path in $\mathbb{G}_2$ we get $c_2(f(s), f(t)) \le \sum_{i=1}^{k} w_2(f(s_{i-1}), f(s_i))$. It follows that $c_2(f(s), f(t)) \le c_1(s, t)$ since $w_2(f(u), f(v)) \le w_1(u, v)$ for all $u, v \in S_1$. Hence, τ is $\mathbf{A}_\downarrow$ since s and t were chosen arbitrarily.

Let $s, t, u \in S_1$ be three arbitrary states. We first note that $c_2(f(s), f(t)) \le c_2(f(s), f(u)) + c_2(f(u), f(t))$ by the definition of cost functions. We also note that $c_2(f(s), f(u)) \le c_1(s, u)$ since τ is $\mathbf{A}_\downarrow$. Hence, $c_2(f(s), f(t)) \le c_2(f(s), f(u)) + c_2(f(u), f(t)) \le c_1(s, u) + c_2(f(u), f(t))$, that is, τ is $\Delta_\downarrow$ since s, t and u were chosen arbitrarily.

2) Let $S_1 = S_2 = \{s_1, s_2\}$, let $E_1 = E_2 = \{\langle s_1, s_2, \ell\rangle\}$, let $f(s) = \{s\}$ for all $s \in S_1$ and let $R = \{\langle \ell, \ell\rangle\}$. Also set $w_1(s_1, s_2) = 1$ and $w_2(s_1, s_2) = 2$. Then τ is $\mathbf{M}_\uparrow\mathbf{R}_\uparrow\mathbf{C}_\uparrow$ but neither $\mathbf{A}_\downarrow$ nor $\Delta_\downarrow$ for this instance. □

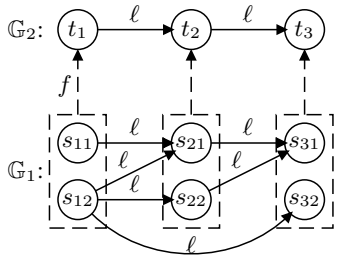

Figure 3: $\mathbf{P_{S\uparrow}}$ but not $\mathbf{A}_\downarrow$.

6 Examples

In this section we give examples of how the extended framework can be used to express new things or express old things in new ways. The examples are deliberately quite different from each other in order to demonstrate the breadth of applicability of an abstract framework of this kind.

Admissibility and Homomorphisms

Homomorphisms, i.e. $\mathbf{M}_\uparrow\mathbf{R}_\uparrow\mathbf{C}_\uparrow$ transformations, are known to be very suitable as abstraction functions in heuristic search, cf. Holte *et al.* [1996a], Helmert *et al.* [2007] and Zilles and Holte [2010]. One reason for this is that they are admissible. The following result shows that also the opposite holds, admissibility implies that the abstraction function is a homomorphism, if all arcs in both graphs have unit cost. Without unit costs, this relationship breaks down. This is a most relevant observation in the context of using the path length in the abstract graph as the heuristic estimate for the path length or path cost in the ground graph. Zero cost arcs are typically not considered in this context.

Theorem 16. *Let $\tau = \langle f, R, w_1, d_2\rangle$ be an $\mathbf{M}_\uparrow\mathbf{A}_\downarrow$ transformation s.t. $d_1(s, t) \le w_1(s, t)$ for all edges $\langle s, t\rangle \in E_1$. Then: 1) f is a homomorphism if $w_1 = d_1$. 2) Otherwise f need not be a homomorphism.*

Proof. We ignore labels since they are irrelevant.

1) Suppose τ is $\mathbf{M}_\uparrow\mathbf{A}_\downarrow$ and $w_1 = d_1$ but f is not a homomorphism. Then there is some $\langle s, t\rangle \in E_1$ s.t. $\langle f(s), f(t)\rangle \notin E_2$. That is, $c_1(s, t) = d_1(s, t) = 1$, but if there is a path from $f(s)$ to $f(t)$, then it must be of length 2 or more. Hence, $c_2(f(s), f(t)) \ge 2$ so τ cannot be $\mathbf{A}_\downarrow$. This contradicts the assumption, so f must be a homomorphism.

2) Let $S_1 = S_2 = \{s_1, s_2, s_3\}$, $E_1 = \{\langle s_1, s_3\rangle\}$, $E_2 = \{\langle s_1, s_2\rangle, \langle s_2, s_3\rangle\}$. Let $f(s) = s$ for all $s \in S_1$ and let $w_1(s_1, s_3) = 2$. Clearly, τ is $\mathbf{A}_\downarrow$ for this example. However, f is not a homomorphism since $\langle s_1, s_3\rangle \in E_1$ but $\langle f(s_1), f(s_3)\rangle \notin E_2$. □

Spurious States

The *downward path preserving (**DPP**)* property [Zilles and Holte, 2010] guarantees that the abstract search space does

not contain any *spurious states* (abstract goals states not corresponding to ground goal states). We have earlier proved that **DPP** is equivalent to $\mathbf{P}_{\updownarrow}$ [Bäckström and Jonsson, 2012a]. However, $\mathbf{A}_{\downarrow} \Rightarrow \mathbf{P}_{\mathbf{S}\uparrow}$ and $\mathbf{P}_{\mathbf{S}\uparrow} \Rightarrow \mathbf{P}_{\uparrow}$, so $\mathbf{P}_{\downarrow}\mathbf{A}_{\downarrow} \Rightarrow$ **DPP**. Although $\mathbf{P}_{\downarrow}\mathbf{A}_{\downarrow}$ is a stronger criterion than **DPP**, it means that we need not verify independently that $\mathbf{P}_{\uparrow}$ holds if we already know that the heuristic is admissible, which is typically the case when considering **DPP**.

Globally Admissible Heuristics

Karpas and Domshlak [2012] considered optimal solutions with non-admissible heuristics. One example is so called *globally admissible heuristics*, which need only be admissible for the states along some optimal plan. Let $\mathbb{G} = \langle S, E \rangle$ be an STG, c a cost function for $\mathbb{G}$ and h a heuristic for c. Then, for arbitrary $s, t \in S$, h is *globally admissible* for c from s to t if there is an optimal path $s_0, \ldots, s_n$ such that $s_0 = s$, $s_n = t$ and $h(s_i, t) \leq c(s_i, t)$ for all i, $0 \leq i \leq n$. This concept can be alternatively characterized as follows.

Theorem 17. *Let $\tau = \langle f, R, w_1, w_2 \rangle$ be an $\boldsymbol{M}_{\uparrow}\boldsymbol{A}_{\boldsymbol{C}\downarrow}$ transformation. Then, if there is an optimal path σ from s to t in $\mathbb{G}_1$ such that $f(\sigma)$ is a path in $\mathbb{G}_2$, then c_2 is a globally admissible heuristic for c_1 from s to t.*

This makes use of properties $\mathbf{M}_{\uparrow}$ and $\mathbf{A}_{\mathbf{C}\downarrow}$ and a type of completeness property that is even weaker than $\mathbf{P}_{\mathbf{T}\uparrow}$.

Valtorta's Theorem

Valtorta [1984] proved that when using embeddings as abstraction functions, it is not possible to explore fewer nodes in total, counting both ground and abstract nodes, when using A* search with path length in the abstract graph as heuristic estimate. This was later generalized to abstraction functions in general, by Holte *et al.* [1996b], known as the generalized version of Valtorta's theorem.

Theorem 18. *[Generalized Valtorta's theorem, [Holte* et al.*, 1996b]] Assume $\tau = \langle f, R, w_1, d_2 \rangle$ is an $\boldsymbol{M}_{\uparrow}$ transformation and $d_1(s,t) \leq w_1(s,t)$ for all $\langle s, t \rangle \in E_1$. Let u be any state in $\mathbb{G}_1$ that is necessarily expanded when the instance $\langle s, t \rangle$ is solved by BFS in $\mathbb{G}_1$ and let the heuristic function h be $h(u,t) = d_2(f(u), f(t))$, computed by BFS in $\mathbb{G}_2$. If A* solves this instance, then either u or $f(u)$ will be expanded during the search.*

Holte et. al. noted that this does not rule out that some abstractions might explore fewer nodes in total. (The theorem is somewhat weak, though, since it only tells us that u *or* $f(u)$ is expanded, allowing the possibility that both are expanded). It is well known that this requires that there are abstract nodes corresponding to two or more ground nodes; the expansion of an abstract node can then result in a heuristic estimate that prevents A* from exploring the corresponding ground nodes. An alternative characterization of this is that f is not $\mathbf{M}_{\downarrow}$. While this is hardly an interesting new result itself, it demonstrates that the framework defined so far is sufficient to express this important criterion for A* search.

We may also step outside the $\mathbf{M}_{\uparrow}$ assumption. Suppose we have some concept of expanding $f(u)$ for a state u when f is not $\mathbf{M}_{\uparrow}$. We need not have a precise definition of this concept to see that Theorem 18 still holds, and that f must still not be $\mathbf{M}_{\downarrow}$ to have any chance of exploring fewer nodes.

7 Discussion

We previously used our framework to model a number of abstraction methods in planning, derive their transformation properties and draw conclusions from that regarding soundness and completeness [Bäckström and Jonsson, 2012a]. We will very briefly sketch how these results together with the new results of this paper can sometimes be used to also say about the metric properties of these abstraction methods.

Method **ABII** (ABSTRIPS in the version where non-critical atoms are removed everywhere, cf. Knoblock [1994]) is known to be an $\mathbf{M}_{\uparrow}\mathbf{R}_{\uparrow}\mathbf{C}_{\uparrow}$ transformation, i.e. a homomorphism. Hence, it is also an $\mathbf{A}_{\downarrow}\Delta_{\downarrow}$ transformation, i.e. it can be used as an admissible and consistent heuristic. Although this is already known, it is interesting that the result can be deduced in this way, using only abstract transformation properties. Methods **RRAa** and **RRAb** are two extreme cases of the concept of removing redundant actions [Haslum and Jonsson, 2000]. It is not hard to see that **RRAa** is $\mathbf{M}_{\uparrow}\mathbf{R}_{\uparrow}\mathbf{C}_{\uparrow}$ and, consequently, $\mathbf{A}_{\downarrow}\Delta_{\downarrow}$ while **RRAb** is neither $\mathbf{A}_{\downarrow}$ nor $\Delta_{\downarrow}$. Method **IDL** is the common abstraction method of ignoring delete lists of actions, and it is $\mathbf{M}_{\updownarrow}\mathbf{R}_{\updownarrow}$ but not $\mathbf{C}_{\uparrow}$ or $\mathbf{C}_{\downarrow}$. Hence, it is not admissible, which might seem to contradict known results. It does not, however, since Bäckström and Jonsson allowed negative preconditions, which is usually not considered when ignoring delete lists. This indicates that the equivalence between STRIPS with and without negative preconditions [Bäckström, 1995] must be taken with care in some cases, like delete relaxation.

Let us finally discuss some future research directions. Our results make no particular assumptions about the function f and the relation R in the transformations. Looking also at restricted cases would be interesting, since the abstraction methods used in the literature impose various restrictions on f and R. The preceeding paragraph gives some ideas about what kind of results could be achieved this way.

The results in Section 5 clearly demonstrate that the lack of metrics in usual refinement concepts makes it hard to prove further positive results on the connections between refinement and heuristics. The obvious way forward would be to somehow add metric aspects also to refinements. The properties $\mathbf{P}_{\mathbf{k}\uparrow}$ and $\mathbf{P}_{\mathbf{k}\downarrow}$ actully do so, but not in a sufficient way. For instance, one might consider a property $\mathbf{P}^m_{k\downarrow}$ that is like $\mathbf{P}_{\mathbf{k}\downarrow}$ but additionally requires that each arc along the path can be refined into a path of length m at most. This might allow for finding tighter relationships between refinement and heuristics, perhaps relating $\mathbf{P}^m_{k\downarrow}$ to approximate heuristic search (eg. using weighted A*). However, it could also provide a deeper insight into refinement itself; it is well known that purely qualitative criteria can cause anomalous behaviour in refinement [Bäckström and Jonsson, 1995].

Another future direction is to study abstractions and heuristics for non-$\mathbf{M}_{\uparrow}$ functions. We note that the literature on this topic is very scarce, and defining such heuristics is much less straightforward. For instance, we can no longer exploit ordinary homomorphisms. One interesting exception is Pang and Holte [2012] who introduce so-called *multimapping abstractions* as a method for aggregating multiple heuristics.

References

[Bäckström and Jonsson, 1995] Christer Bäckström and Peter Jonsson. Planning with abstraction hierarchies can be exponentially less efficient. In *Proc. 14th Int'l Joint Conf. Artif. Intell. (IJCAI'95), Montreal, Canada*, pages 1599–1605, 1995.

[Bäckström and Jonsson, 2012a] Christer Bäckström and Peter Jonsson. Abstracting abstraction in search with applications to planning. In *Proc. 13th Int'l Conf. Knowledge Repr. Reasoning (KR'12), Rome Italy*, pages 446–456, 2012.

[Bäckström and Jonsson, 2012b] Christer Bäckström and Peter Jonsson. Abstracting abstraction in search II: Complexity analysis. In *Proc. 5th Ann. Symp. Combinatorial Search (SoCS'12), Niagara Falls, ON, Canada*, pages 10–17, 2012.

[Bäckström, 1995] Christer Bäckström. Expressive equivalence of planning formalisms. *Artif. Intell.*, 76(1-2):17–34, 1995.

[Clarke *et al.*, 2003] Edmund M. Clarke, Orna Grumberg, Somesh Jha, Yuan Lu, and Helmut Veith. Counterexample-guided abstraction refinement for symbolic model checking. *J. ACM*, 50(5):752–794, 2003.

[Culberson and Schaeffer, 1998] Joseph C. Culberson and Jonathan Schaeffer. Pattern databases. *Computational Intelligence*, 14(3):318–334, 1998.

[Dechter and Pearl, 1985] Rina Dechter and Judea Pearl. Generalized best-first search strategies and the optimality of A*. *J. ACM*, 32(3):505–536, 1985.

[Gregory *et al.*, 2011] Peter Gregory, Derek Long, Craig McNulty, and Susan M. Murphy. Exploiting path refinement abstraction in domain transition graphs. In *Proc. 25th AAAI Conf. Artif. Intell. (AAAI'11), San Francisco, CA, USA*, pages 971–976, 2011.

[Haslum and Jonsson, 2000] Patrik Haslum and Peter Jonsson. Planning with reduced operator sets. In *Proc. 5th Int'l Conf. on Artif. Intell. Planning Systems (AIPS'00), Breckenridge, CO, USA*, pages 150–158, 2000.

[Haslum *et al.*, 2007] Patrik Haslum, Adi Botea, Malte Helmert, Blai Bonet, and Sven Koenig. Domain-independent construction of pattern database heuristics for cost-optimal planning. In *Proc. 17th Int'l Conf. Automated Planning and Scheduling (ICAPS'07), Providence, RI, USA*, pages 1007–1012, 2007.

[Helmert and Domshlak, 2009] Malte Helmert and Carmel Domshlak. Landmarks, critical paths and abstractions: What's the difference anyway? In *Proc. 19th Int'l Conf. Automated Planning and Scheduling (ICAPS'09), Thessaloniki, Greece*, pages 162–169, 2009.

[Helmert *et al.*, 2007] Malte Helmert, Patrik Haslum, and Jörg Hoffmann. Flexible abstraction heuristics for optimal sequential planning. In *Proc. 17th Int'l Conf. Automated Planning and Scheduling (ICAPS'07), Providence, RI, USA*, pages 176–183, 2007.

[Helmert, 2006] Malte Helmert. The fast downward planning system. *J. Artif. Intell. Res.*, 26:191–246, 2006.

[Holte *et al.*, 1996a] Robert C. Holte, T. Mkadmi, Robert M. Zimmer, and Alan J. MacDonald. Speeding up problem solving by abstraction: A graph oriented approach. *Artif. Intell.*, 85(1-2):321–361, 1996.

[Holte *et al.*, 1996b] Robert C. Holte, M. B. Perez, Robert M. Zimmer, and Alan J. MacDonald. Hierarchical A*: Searching abstraction hierarchies efficiently. In *Proc. 13th Nat'l Conf. Artif. Intell. (AAAI'96), Portland, OR, USA, Vol. 1.*, pages 530–535, 1996.

[Karpas and Domshlak, 2012] Erez Karpas and Carmel Domshlak. Optimal search with inadmissible heuristics. In *Proc. 22nd Int'l Conf. Automated Planning and Scheduling (ICAPS'12), Atibaia, São Paulo, Brazil*, pages 92–100, 2012.

[Knoblock, 1994] Craig A. Knoblock. Automatically generating abstractions for planning. *Artif. Intell.*, 68(2):243–302, 1994.

[Newell *et al.*, 1959] Allen Newell, J. C. Shaw, and Herbert A. Simon. Report on a general problem-solving program. In *IFIP Congress, Paris, France*, pages 256–264. UNESCO, 1959.

[Pang and Holte, 2012] B. Pang and R. Holte. Multimapping abstractions and hierarchical heuristic search. In *Proc. 5th Ann. Symp. Combinatorial Search (SoCS'12), Niagara Falls, ON, Canada*, pages 72–79, 2012.

[Sacerdoti, 1974] Earl D. Sacerdoti. Planning in a hierarchy of abstraction spaces. *Artif. Intell.*, 5(2):115–135, 1974.

[Seipp and Helmert, 2013] Jendrik Seipp and Malte Helmert. Counterexample-guided cartesian abstraction refinement. In *Proc. 23rd Int'l Conf. Automated Planning and Scheduling, (ICAPS'13), Rome, Italy*, 2013.

[Sturtevant and Buro, 2005] Nathan R. Sturtevant and Michael Buro. Partial pathfinding using map abstraction and refinement. In *Proc. 20th Nat'l Conf. Artif. Intell. (AAAI'05), Pittsburgh, PA, USA*, pages 1392–1397, 2005.

[Valtorta, 1984] Marco Valtorta. A result on the computational complexity of heuristic estimates for the A* algorithm. *Inf. Sci.*, 34(1):47–59, 1984.

[Yang *et al.*, 2008] Fan Yang, Joseph C. Culberson, Robert Holte, Uzi Zahavi, and Ariel Felner. A general theory of additive state space abstractions. *J. Artif. Intell. Res. (JAIR)*, 32:631–662, 2008.

[Zilles and Holte, 2010] Sandra Zilles and Robert C. Holte. The computational complexity of avoiding spurious states in state space abstraction. *Artif. Intell.*, 174(14):1072–1092, 2010.