Symmetry Breaking in Deterministic Planning as Forward Search: Orbit Space Search Algorithm

Technical Report IS/IE-2015-02

Carmel Domshlak
DCARMEL@IE.TECHNION.AC.IL
Technion, Haifa, Israel

Michael Katz
KATZM@IL.IBM.COM
IBM Research Lab, Haifa, Israel

Alexander Shleyfman
ALESH@TX.TECHNION.AC.IL
Technion, Haifa, Israel

Abstract

This technical report describes the orbit space search algorithm that is used in the *Metis* planner, a participant of the International Planning Competition (IPC) 2014 (Alkhazraji, Katz, Mattmüller, Pommerening, Shleyfman, & Wehrle, 2014).

1. Introduction

Over the last two decades, the combined machinery of domain-independent heuristics, preferred operators, and various enhancements of the search infrastructure, have positioned heuristic forward search as a leading techniques for deterministic domain-independent planning, in terms of both efficiency and robustness. Numerous admissible and non-admissible heuristics for domain-independent planning have been proposed, varying from cheap to compute and not very informative to expensive to compute and very informative (Bonet & Geffner, 1998, 2001; Haslum & Geffner, 2000; Hoffmann & Nebel, 2001; Helmert, 2004; Helmert, Haslum, & Hoffmann, 2007; Katz & Domshlak, 2010; Karpas & Domshlak, 2009; Helmert & Domshlak, 2009; Bonet & Helmert, 2010; Richter & Westphal, 2010). However, while further progress in developing informative heuristics is still very much desired, both cost-optimal and satisficing searches will unavoidably expand an exponential number of nodes on many problems, even if equipped with heuristics that are almost perfect in their estimates (Pearl, 1984; Helmert & Röger, 2008).

One major reason for this Achilles heel of state-space search is state symmetries in the transition systems of interest. A succinct description of the planning tasks in languages such as STRIPS (Fikes & Nilsson, 1971) and SAS$^+$ (Bäckström & Klein, 1991; Helmert, 2009) almost unavoidably results in many different states in the search space being symmetric to one another with respect to the task at hand. In turn, failing to detect and account for these symmetries results in forward search algorithms, such as A^*, WA^* and greedy best-first search (*GBFS*), searching through many symmetric states, despite the fact that searching through a state is equivalent to searches through all of its symmetric counterparts.

Since symmetries show up in probably any type of succinctly specified combinatorial problems and, if not properly taken care of, affect the effectiveness of the search algorithms, the idea of identifying and pruning symmetries while reasoning about automorphisms of the search spaces has been exploited for quite a while already in model checking (Emerson & Sistla, 1996), constraint satisfaction (Puget, 1993), and planning as constrain satisfaction (Rintanen, 2003; Fox & Long, 1999, 2002). More recently, Pochter, Zohar, and Rosenschein (2011) developed a first computational framework for exploiting state-space automorphisms in domain-independent planning as heuristic forward search, and empirically showed the effectiveness of this framework in scaling up A^*-based cost-optimal planning. Continuing the investigation in the direction suggested by Pochter et al. , Domshlak, Katz, and Shleyfman (2012) introduced a sound and complete optimal search algorithm (hereafter referred to as DKS). Unfortunately, both algorithms suffer from high memory consumption due to the need to keep an additional state per search node. In this work we show how this burden can be eliminated.

2. Background

In order to allow for symmetry elimination in A^* search, Domshlak et al. (2012) introduced a sound and complete optimal search algorithm (hereafter referred to as DKS, depicted in Figure 1). DKS extends the duplicate elimination mechanism of A^* to consider symmetrical states as duplicates. Two states are considered to be duplicates if they are mapped to the same canonical state. Similarly to the regular A^* search algorithm, each search node in DKS refers to the state it is associated with, to the parent state of that state, and to the corresponding achieving operator, as well as keeps two numeric values for the heuristic estimate h and the cost-so-far g from the initial state s_0. However, to support the aforementioned extended reasoning about duplicates, each node additionally refers to the canonical state of its state, a data that is not stored by A^*.

Unfortunately, using such duplicate elimination comes not without a price. In cases when reopening is performed, the parent relation might be updated in a way that forces it to lose the connectivity property. Thus, even if a goal state is reached, it is no longer possible to retrace a path from the goal state to the initial state following the parent relation. Fortunately, exploiting information about the symmetries allows to reconstruct such a path in a way described by the TRACE-FORWARD procedure.

3. Orbit State Search

Naturally, the aforementioned requirement of storing an additional state per node for duplicate elimination is a considerable disadvantage since it results in substantial increase in memory consumption. To overcome this disadvantage, we introduce a simple modification of the DKS search algorithm, called *orbit search*. In contrast to DKS, *only* the canonical state is stored per node. These canonical states define *orbits* of (symmetric) states that have the same canonical state – hence the name orbit search. Note that the plan reconstruction procedure of DKS makes the implementation of orbit search extremely simple. There is only a minor, almost syntactic, difference between A^* and orbit search, which is that the states are replaced by their canonical representatives when stored

```
Let $\Sigma :=$ subset of generators for the group $\Gamma_{S_*}$.
Let $\mathcal{C} : S \rightarrow S$ the "canonical states" mapping based on $\Sigma$
Let node be a tuple $\langle$state, parent, operator, $g$, heuristic, canonical$\rangle$
$OPEN \leftarrow \{\langle s_0, \emptyset, \emptyset, 0, 0, \mathcal{C}(s_0)\rangle\}$, $CLOSED \leftarrow \emptyset$
while $OPEN \neq \emptyset$ do
    Remove best node $n$ from $OPEN$
    $CLOSED \leftarrow CLOSED \cup \{n\}$
    $s \leftarrow n$.state
    if $s \in S_*$ then return TRACE-FORWARD(TRACE($n$),$\Gamma_{S_*}$)
    // EXPAND $s$
    for all applicable operators $o$ of $s$ do
        $s' \leftarrow s[\![o]\!]$
        $g' \leftarrow n.g + C(o)$
        if $\exists n' \in CLOSED \cup OPEN$ such that $n'$.canonical $= \mathcal{C}(s')$ then
            if $g' < n'.g$ then
                // REOPEN $n'$
                $CLOSED \leftarrow CLOSED \setminus \{n'\}$
                $OPEN \leftarrow (OPEN \setminus \{n'\}) \cup \{\langle n'$.state, $s, o, g', n'$.heuristic, $n'$.canonical$\rangle\}$
        else
            $h \leftarrow$ EVALUATE($s'$)
            $OPEN \leftarrow OPEN \cup \{\langle s', s, o, g', h, \mathcal{C}(s')\rangle\}$
return NO SOLUTION

procedure TRACE-FORWARD($\pi' = \langle(\varepsilon, s'_0), (o'_1, s'_1), \ldots, (o'_m, s'_m)\rangle, \Gamma$)
    Let $\sigma_0 \in \Gamma$ be such that $\sigma_0(s'_0) = s_0$
    $\pi \leftarrow \langle(\varepsilon, s_0)\rangle$
    for $i := 1$ to $m$ do
        Let $\sigma \in \Gamma$ be such that $\sigma(s'_i) = s'_{i-1}[\![o'_i]\!]$
        $\sigma_i \leftarrow \sigma \circ \sigma_{i-1}$
        $s_i \leftarrow \sigma_i(s'_i)$
        Let $o_i$ be a cheapest operator such that $s_{i-1}[\![o_i]\!] = s_i$
        $\pi \leftarrow \pi \cdot \langle(o_i, s_i)\rangle$
return $\pi$
```

Figure 1: The *DKS* adaptation of A^*. TRACE(n) is the regular solution trace procedure of A^*.

in the OPEN and CLOSED lists. Once a goal state is found, the plan reconstruction is performed as in *DKS*.

The orbit search modification of the A^* algorithm is depicted in Figure 2. Note that the role of the successor state s' in the A^* algorithm is delegated here to its canonical representative s'_c and the duplicate detection of A^* will actually compare two nodes by their respective canonical states.

```
Let $\Sigma :=$ subset of generators for the group $\Gamma_{S_*}$.
Let $\mathcal{C} : S \to S$ the "canonical states" mapping based on $\Sigma$
Let node be a tuple $\langle \mathsf{state}, \mathsf{parent}, \mathsf{operator}, g, \mathsf{heuristic} \rangle$
$\mathit{OPEN} \leftarrow \{\langle s_0, \emptyset, \emptyset, 0, 0 \rangle\}$, $\mathit{CLOSED} \leftarrow \emptyset$
while $\mathit{OPEN} \neq \emptyset$ do
    Remove best node $n$ from $\mathit{OPEN}$
    $\mathit{CLOSED} \leftarrow \mathit{CLOSED} \cup \{n\}$,    $s \leftarrow n.\mathsf{state}$
    if $s \in S_*$ then return TRACE-FORWARD(TRACE($n$),$\Gamma_{S_*}$)
    // EXPAND $s$
    for all applicable operators $o$ of $s$ do
        $s' \leftarrow s[\![o]\!]$,    $g' \leftarrow n.g + C(o)$
        $s'_c \leftarrow \mathcal{C}(s')$
        if $\exists n' \in \mathit{CLOSED} \cup \mathit{OPEN}$ such that $n'.\mathsf{state} = s'_c$ then
            if $g' < n'.g$ then
                // REOPEN $n'$
                $\mathit{CLOSED} \leftarrow \mathit{CLOSED} \setminus \{n'\}$
                $\mathit{OPEN} \leftarrow (\mathit{OPEN} \setminus \{n'\}) \cup \{\langle n'.\mathsf{state}, s, o, g', n'.\mathsf{heuristic} \rangle\}$
        else
            $h \leftarrow$ EVALUATE($s'_c$)
            $\mathit{OPEN} \leftarrow \mathit{OPEN} \cup \{\langle s'_c, s, o, g', h \rangle\}$
return NO SOLUTION
```

Figure 2: The *OS* search algorithm—the "orbit search" variant of *DKS*.

4. Summary

We introduced an extension of the A^* algorithm that preserves its core properties of completeness and optimality. Similarly to the works of Pochter et al. (2011) and Domshlak et al. (2012) our approach works at the level of the search algorithm, and is completely independent of the heuristic in use. Furthermore, our approach allows for completely eliminating the requirement of both previous algorithms for storing an additional state in each search node, leading to better memory consumption.

References

Alkhazraji, Y., Katz, M., Mattmüller, R., Pommerening, F., Shleyfman, A., & Wehrle, M. (2014). Metis: Arming Fast Downward with pruning and incremental computation. In *Eighth International Planning Competition (IPC-8): planner abstracts*, pp. 88–92.

Bäckström, C., & Klein, I. (1991). Planning in polynomial time: the SAS-PUBS class. *Computational Intelligence*, *7*(3), 181–197.

Bonet, B., & Geffner, H. (1998). HSP: Heuristic search planner. In *AIPS'98 Planning Competition*, Pittsburgh, PA.

Bonet, B., & Geffner, H. (2001). Planning as heuristic search. *Artificial Intelligence*, *129*(1), 5–33.

Bonet, B., & Helmert, M. (2010). Strengthening landmark heuristics via hitting sets. In Coelho, H., Studer, R., & Wooldridge, M. (Eds.), *Proceedings of the 19th European Conference on Artificial Intelligence (ECAI 2010)*, pp. 329–334. IOS Press.

Domshlak, C., Katz, M., & Shleyfman, A. (2012). Enhanced symmetry breaking in cost-optimal planning as forward search. In McCluskey, L., Williams, B., Silva, J. R., & Bonet, B. (Eds.), *Proceedings of the Twenty-Second International Conference on Automated Planning and Scheduling (ICAPS 2012)*. AAAI Press.

Emerson, E. A., & Sistla, A. P. (1996). Symmetry and model checking. *Formal Methods in System Design*, *9*(1–2), 105–131.

Fikes, R. E., & Nilsson, N. J. (1971). STRIPS: A new approach to the application of theorem proving to problem solving. *Artificial Intelligence*, *2*, 189–208.

Fox, M., & Long, D. (1999). The detection and exploitation of symmetry in planning problems. In Dean, T. (Ed.), *Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence (IJCAI 1999)*, pp. 956–961. Morgan Kaufmann.

Fox, M., & Long, D. (2002). Extending the exploitation of symmetries in planning. In Ghallab, M., Hertzberg, J., & Traverso, P. (Eds.), *Proceedings of the Sixth International Conference on Artificial Intelligence Planning and Scheduling (AIPS 2002)*, pp. 83–91. AAAI Press.

Haslum, P., & Geffner, H. (2000). Admissible heuristics for optimal planning. In Chien, S., Kambhampati, S., & Knoblock, C. A. (Eds.), *Proceedings of the Fifth International Conference on Artificial Intelligence Planning and Scheduling (AIPS 2000)*, pp. 140–149. AAAI Press.

Helmert, M. (2004). A planning heuristic based on causal graph analysis. In Zilberstein, S., Koehler, J., & Koenig, S. (Eds.), *Proceedings of the Fourteenth International Conference on Automated Planning and Scheduling (ICAPS 2004)*, pp. 161–170. AAAI Press.

Helmert, M. (2009). Concise finite-domain representations for PDDL planning tasks. *Artificial Intelligence*, *173*, 503–535.

Helmert, M., & Domshlak, C. (2009). Landmarks, critical paths and abstractions: What's the difference anyway?. In Gerevini, A., Howe, A., Cesta, A., & Refanidis, I. (Eds.), *Proceedings of the Nineteenth International Conference on Automated Planning and Scheduling (ICAPS 2009)*, pp. 162–169. AAAI Press.

Helmert, M., Haslum, P., & Hoffmann, J. (2007). Flexible abstraction heuristics for optimal sequential planning. In Boddy, M., Fox, M., & Thiébaux, S. (Eds.), *Proceedings of the Seventeenth International Conference on Automated Planning and Scheduling (ICAPS 2007)*, pp. 176–183. AAAI Press.

Helmert, M., & Röger, G. (2008). How good is almost perfect?. In *Proceedings of the Twenty-Third AAAI Conference on Artificial Intelligence (AAAI 2008)*, pp. 944–949. AAAI Press.

Hoffmann, J., & Nebel, B. (2001). The FF planning system: Fast plan generation through heuristic search. *Journal of Artificial Intelligence Research*, *14*, 253–302.

Karpas, E., & Domshlak, C. (2009). Cost-optimal planning with landmarks. In *Proceedings of the 21st International Joint Conference on Artificial Intelligence (IJCAI 2009)*, pp. 1728–1733.

Katz, M., & Domshlak, C. (2010). Implicit abstraction heuristics. *Journal of Artificial Intelligence Research*, *39*, 51–126.

Pearl, J. (1984). *Heuristics: Intelligent Search Strategies for Computer Problem Solving*. Addison-Wesley.

Pochter, N., Zohar, A., & Rosenschein, J. S. (2011). Exploiting problem symmetries in state-based planners. In Burgard, W., & Roth, D. (Eds.), *Proceedings of the Twenty-Fifth AAAI Conference on Artificial Intelligence (AAAI 2011)*, pp. 1004–1009. AAAI Press.

Puget, J.-F. (1993). On the satisfiability of symmetrical constrained satisfaction problems. In *Proceedings of the 7th International Symposium on Methodologies for Intelligent Systems (ISMIS 1993)*, pp. 350–361.

Richter, S., & Westphal, M. (2010). The LAMA planner: Guiding cost-based anytime planning with landmarks. *Journal of Artificial Intelligence Research*, *39*, 127–177.

Rintanen, J. (2003). Symmetry reduction for SAT representations of transition systems. In Giunchiglia, E., Muscettola, N., & Nau, D. (Eds.), *Proceedings of the Thirteenth International Conference on Automated Planning and Scheduling (ICAPS 2003)*, pp. 32–40. AAAI Press.