SymPA: Symbolic Perimeter Abstractions for Proving Unsolvability

Álvaro Torralba
Saarland University
Saarbrücken, Germany
torralba@cs.uni-saarland.de

Abstract

This paper describes the SymPA planner that participated in the 2016 unsolvability International Planning Competition (IPC). SymPA is built on top of SymBA*, the winner of the optimal-track of the 2014 IPC. SymBA* combines symbolic bidirectional search and perimeter abstraction heuristics. However, despite to the close relation between cost-optimal planning and deciding plan existence, SymBA* is not suitable for proving unsolvability. SymPA is the result of tailoring symbolic perimeter abstraction heuristics for proving unsolvability of planning problems.

Introduction

Symbolic search is a technique for state-space exploration that uses efficient data-structures, usually Binary Decision Diagrams (BDDs) (Bryant 1986), to represent and manipulate sets of states. This has been a successful approach in different areas in which algorithms must exhaustively search the state space like model-checking (McMillan 1993), planning with uncertainty (Cimatti and Roveri 2000) or classical planning. The pioneer in classical planning was the MIPS planner (Edelkamp and Helmert 2001) that used bidirectional search. Symbolic search was also used for computing Pattern Database (PDB) heuristics (Edelkamp 2001; 2002). The Gamer planner popularized the use of bidirectional search and BDDA* with symbolic pattern databases (Kissmann and Edelkamp 2011; Kissmann 2012). Moreover, recent improvements (Torralba *et al.* 2013a; Torralba and Alcázar 2013) have posed symbolic bidirectional search as a state-of-the-art algorithm for cost-optimal planning. A clear representative of this trend is the SymBA* planner, which uses symbolic perimeter abstractions to inform a symbolic bidirectional A* search (Torralba *et al.* 2016) and won the optimal-track of IPC-14.

Proving unsolvability via search requires to completely exhaust the state space so symbolic search is very promising. Moreover, abstractions are specially useful in unsolvable problems since it suffices to find an unsolvable abstract problem (Bäckström *et al.* 2013). However, SymBA* is a cost-optimal planner so it focuses on searching plans of lower cost. In this paper, we present SymPA (standing for symbolic perimeter abstractions) that tailors symbolic search and perimeter PDBs for proving unsolvability.

Symbolic Perimeter Abstractions

Abstraction heuristics map the state space into a smaller abstract state space and use the optimal solution cost as an estimation for the original problem. There are different types of abstraction heuristics depending on how the mapping is defined. Pattern Databases (PDBs) (Culberson and Schaeffer 1998; Edelkamp 2001) are projections of the planning task onto a subset of variables (called *pattern*), so that two states are equivalent iff they agree on the value of variables in the pattern. Merge-and-shrink (M&S) abstractions generalize PDBs, allowing to derive abstractions that use all variables (Helmert *et al.* 2007; 2014).

Here we focus on PDBs so the abstraction mapping is determined by a subset of variables, $W \subseteq \mathcal{V}$. We will use $\mathcal{T}^W$ to denote a search that takes into account variables in W and ignore the rest. Hence, $\mathcal{T}^{\mathcal{V}}$ represents a search on the state space of the planning task. To distinguish the direction of the search, we will use $\mathcal{T}^W_{fw}$ and $\mathcal{T}^W_{bw}$ to refer to searches in the forward and backward direction, respectively. We denote a search in an unspecified direction by $\mathcal{T}^W_u$.

Perimeter abstractions construct a perimeter around the goal in the original state space and use it to seed the search in the abstract state space (Felner and Ofek 2007; Eyerich and Helmert 2013). The perimeter is constructed by a backward search, $\mathcal{T}^{\mathcal{V}}_{bw}$, which computes the perfect heuristic for all states in $closed(\mathcal{T}^{\mathcal{V}}_{bw})$. For states outside the perimeter, an abstract search, $\mathcal{T}^W_{bw}$ computes the minimum distance from each abstract state to the abstract perimeter.

Symbolic perimeter abstractions generalized this idea, introducing the use of multiple levels of abstractions (Torralba *et al.* 2013b; Torralba 2015). Contrary to other PDB approaches that start from single-variable PDBs and iteratively add more variables into the pattern (Haslum *et al.* 2007; Bäckström *et al.* 2013), symbolic perimeter PDBs aim to relax the search as little as possible. Hence, they start building a perimeter that considers all variables and, only when the search is unfeasible, remove variables from the pattern one by one until the search can be continued.

The SPM&S planner computes several perimeter M&S and PDB heuristics in backward direction to inform an A* search. SPM&S participated in the cost-optimal track of IPC14 and was competitive with other heuristic search planning, only behind of symbolic bidirectional search planners.

SymBA*: Bidirectional Search with Perimeter Abstraction Heuristics

SymBA* performs several symbolic bidirectional A* searches on different state spaces. First, SymBA* starts a bidirectional search in the original state space, $\mathcal{T}^{\mathcal{V}}$. At each iteration, the algorithm performs a step in a selected direction, i.e. expands the set of states with minimum f-value in the frontier. Since no abstraction heuristic has been derived yet, it behaves like symbolic bidirectional uniform-cost search. This search continues until the next layer in both directions is deemed as unfeasible, because SymBA* estimates that it will take either too much time or memory. Only then, a new bidirectional search is started in an abstract state space, $\mathcal{T}^{W}$, $W \subset \mathcal{V}$, initialized with the current frontiers of $\mathcal{T}^{\mathcal{V}}$. The abstract searches provide heuristic estimations, increasing the f-value of states in the original search frontiers. Eventually, the search in the original state space will be simplified and it will be continued.

The overall strategy of SymBA* is motivated by the great results of symbolic bidirectional uniform-cost search in cost-optimal planning. Therefore, SymBA* is configured to run as much search as possible in the original state space, only resorting to use abstractions when the unabstracted search becomes unfeasible. Hence, most theory of SymBA* is devoted to how evaluate the heuristics in a lazy way, minimizing the amount of search performed in abstract state spaces.

However, SymBA* is not completely suitable for proving unsolvability for several reasons:

1) Bidirectional search is less effective In problems with a solution bidirectional search is effective because, instead of searching a direction until the depth of the optimal solution d, we perform two searches until depth $d/2$. This is a great advantage since the number of explored states often grows exponentially in the search depth. However, unsolvability will only be proved whenever one of the frontiers is completely exhausted. Hence, performing a forward and a backward search just duplicates the planner effort.

2) Abstractions are more effective In cost-optimal planning, doing an abstract search cannot possibly solve the problem. However, in the unsolvability case, often a subset of variables is enough to prove unsolvability so it is possible to prove unsolvability without doing any search in the original state space. In the experimental analysis done by Torralba *et al.* (2016), it was shown that SymBA* does not always benefits from abstraction heuristics because it is hard to find good abstractions that simplify the search while preserving goal-distance information. However, as analyzed by (Bäckström *et al.* 2013), often considering a (sometimes small) subset of variables is enough to prove unsolvability. Hence, searching small abstract state spaces can be expected to be much more effective in problems without any solution.

3) Costs are ignored Action-costs, as well as distance to the goal and/or the initial state are irrelevant in order to prove unsolvability. This, allows to greatly simplify most parts of the planner, such as ignoring the g-value of states and the heuristic evaluation.

SymPA's Algorithm

As SymBA*, SymPA performs searches in forward and backward direction in the original and abstract state spaces. However, there are a number of differences with respect to SymBA*. First of all, since the cost of the path is not relevant, symbolic breadth-first search is used instead of uniform-cost or BDDA*. This simplifies the representation of the open list because a single BDD is used to contain all the states in the frontier, instead of separated BDDs for different g-values.

Moreover, despite it performs searches in both directions, searches in SymPA are not truly bidirectional because there is no direct interaction between the two frontiers, i.e. the planner does not need to check whether the frontiers intersect. The planner benefits from doing searches in both directions because of two reasons. On the one hand, forward and backward search may have different performance so by doing both, the planner benefits of always using the best one for the problem at hand, like portfolio approaches. On the other hand, both directions have synergy when using abstractions since states that are unreachable in one direction are dead-ends in the opposite one.

Finally, while SymBA* fosters the bidirectional search in the original state space until it is completely unfeasible, SymPA performs multiple searches in smaller abstract state spaces in order to discover dead-ends that will help with searches in larger state spaces.

Algorithm 1 shows the pseudocode of SymPA. SymPA maintains a set of ongoing searches, $SearchPool$, that is initialized to searches in both directions in the original state space. It also keeps a set of dead-end states for forward, D_{fw}, and backward, D_{fw}, search that are initialized empty. A search is considered to be feasible if its frontier is represented with less than M BDD nodes, where M is a parameter of the algorithm. M is dynamically adjusted, starting in a relatively low value to explore different abstract searches and increased over time in order to explore less relaxed state spaces. At every iteration, if any search in the pool is feasible, the search deemed as easiest (according to a time estimation based on the time taken by the last step and the number of BDD nodes used to represent its frontier) is continued one more step.

Whenever a search is terminated, if it did not find any solution, the problem has been proved unsolvable and the algorithm terminates. If the search in the original state space ($\mathcal{T}_u^{\mathcal{V}}$) found a solution, then the problem has been proved solvable. Otherwise, we eliminate the search from the pool of ongoing searches, gather all unreachable states (which are dead-ends for the opposite direction) and remove them from all searches in the opposite direction.

When all current searches are unfeasible, a symbolic perimeter abstraction is constructed. We randomly select the forward or backward direction and start from the perimeter on the original state space. The search is relaxed by abstracting some variables away, until it becomes feasible. The process interleaves abstraction and search steps until the search is finished, storing intermediate results in the search pool to be continued later if they become feasible.

Algorithm 1: SymPA

Input: Planning problem: $\Pi = \langle \mathcal{V}, \mathcal{A}, \mathcal{I}, \mathcal{G} \rangle$
Output: "Solvable" or "Unsolvable"

$SearchPool \leftarrow \{\mathcal{T}^{\mathcal{V}}_{fw}, \mathcal{T}^{\mathcal{V}}_{bw}\}$;
$D_{fw}, D_{bw} \leftarrow \emptyset, \emptyset$;
Loop
  if $\exists \mathcal{T}^X_u \in SearchPool$ *s.t.* `Is-Feasible(`$\mathcal{T}^X_u$`)` **then**
    $\mathcal{T}^X_u \leftarrow$ `Easiest-Search(`$SearchPool$`)` ;
    `Expand-frontier(`$\mathcal{T}^X_u, D_u$`)` ;
  else
    $\mathcal{T}^X_u \leftarrow$ `RandomSelection(`$\{\mathcal{T}^{\mathcal{V}}_{fw}, \mathcal{T}^{\mathcal{V}}_{bw}\}$`)` ;
    while $\mathcal{T}^X_u$ *is not finished* **do**
      if `Is-Feasible(`$\mathcal{T}^X_u$`)` **then**
        `Expand-frontier(`$\mathcal{T}^X_u, D_u$`)` ;
      else
        $SearchPool \leftarrow SearchPool \cup \{\mathcal{T}^X_u\}$;
        $\mathcal{T}^X_u \leftarrow$ `Relax-frontier(`$\mathcal{T}^X_u$`)` ;
  if $\mathcal{T}^X_u$ *is finished* **then**
    if not `Found-Solution` *(*$\mathcal{T}^X_u$*)* **then**
      return "Unsolvable" ;
    if $X = V$ **then**
      return "Solvable" ;
    $SearchPool \leftarrow SearchPool \setminus \{\mathcal{T}^X_u\}$;
    $D_{\neg u} \leftarrow D_{\neg u} \cup$ `Unreachable-States(`$\mathcal{T}^X_u$`)` ;
    `Remove-DeadEnds(`$D_{\neg u}, \mathcal{T}^{\Theta}_{\neg u}$`)`;

Abstraction Strategy

The abstraction selection strategy is decisive for the overall performance. Symbolic perimeter PDBs start considering all variables and relax one variable at a time, until the BDD representation of the search frontier has been simplified. To pick which variable to relax, we generate a set of candidate patterns in the following way. Given the pattern to relax, W, we have a candidate for each variable $v_i \in W$, $W_i := W \setminus \{v_i\}$. As pointed out by Haslum *et al.* (2007), patterns that do not contain any goal variable or whose variables form more than one connected component in the causal graph can be ignored because they are not more informed than patterns with strictly less variables. Even though this does not always holds when using perimeter abstractions (because the perimeter might induce a relation between the variables), it is still a good heuristic criterion to focus on useful patterns. Hence, we eliminate candidates that do not contain any goal variable. If a candidate pattern consists of multiple disconnected components in the causal graph, we consider each of them an independent candidate. Finally, we discard all patterns that are subsets of other candidates, in order to relax the search the least possible. Among all candidates, we prefer those that have not previously been selected and pick one of them at random.

IPC configuration

SymPA is built on top of the Fast Downward Planning System (Helmert 2006) and uses h^2 relevance analysis in order to eliminate operators and simplify the planning task prior to the search (Alcázar and Torralba 2015). We submit two different configurations SymPA and SymPA-irr. Both use the procedure described in this paper. The maximum number of BDD nodes for a search to be feasible, M, is set to 10 000 at the beginning. Then, after 300 seconds, is incremented by 10 000 nodes every second. This strategy guarantees that, at the beginning, many variable subsets will be tried and, at the end, the planner will focus on less abstract state spaces using the information discovered by the previous runs.

The only difference between SymPA and SymPA-irr is that the latter also uses a simulation-based irrelevance analysis in order to eliminate operators and simplify the planning task prior to the search (Torralba and Kissmann 2015). This irrelevance analysis constructs a set of transition systems by using M&S with the DFP merge strategy (Dräger *et al.* 2006; Sievers *et al.* 2014) and bisimulation shrinking (Nissim *et al.* 2011) with a limit of 50 000 transitions. Then, it computes a label-dominance simulation relation (Torralba and Hoffmann 2015), which is used to eliminate transitions that can be proved unnecessary to reach the goal. All actions whose transitions are removed this way can be removed from the planning task without affecting plan existence.

Conclusions

This paper has presented the SymPA and SymPA-irr planners that participated in the 2016th edition of the unsolvability IPC. They adapt for proving unsolvability the symbolic bidirectional search and perimeter abstractions techniques successfully used in cost-optimal planning.

Acknowledgements I'd like to thank Jörg Hoffmann and Peter Kissmann for their contributions on irrelevance pruning in SymPA-irr, and the Fast Downward Planning System development team for sharing its latest version.

References

Vidal Alcázar and Álvaro Torralba. A reminder about the importance of computing and exploiting invariants in planning. In Ronen Brafman, Carmel Domshlak, Patrik Haslum, and Shlomo Zilberstein, editors, *Proceedings of the 25th International Conference on Automated Planning and Scheduling (ICAPS'15)*. AAAI Press, 2015.

Christer Bäckström, Peter Jonsson, and Simon Ståhlberg. Fast detection of unsolvable planning instances using local consistency. In Malte Helmert and Gabriele Röger, editors, *Proceedings of the 6th Annual Symposium on Combinatorial Search (SOCS'13)*, pages 29–37. AAAI Press, 2013.

Randal E. Bryant. Graph-based algorithms for boolean function manipulation. *IEEE Transactions on Computers*, 35(8):677–691, 1986.

Alessandro Cimatti and Marco Roveri. Conformant planning via symbolic model checking. *Journal of Artificial Intelligence Research*, 2000.

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

Klaus Dräger, Bernd Finkbeiner, and Andreas Podelski. Directed model checking with distance-preserving abstractions. In Antti Valmari, editor, *Proceedings of the 13th International SPIN Workshop (SPIN 2006)*, volume 3925 of *Lecture Notes in Computer Science*, pages 19–34. Springer-Verlag, 2006.

Stefan Edelkamp and Malte Helmert. MIPS: The model checking integrated planning system. *AI Magazine*, 22(3):67–71, 2001.

Stefan Edelkamp. Planning with pattern databases. In A. Cesta and D. Borrajo, editors, *Proceedings of the 6th European Conference on Planning (ECP'01)*, pages 13–24. Springer-Verlag, 2001.

Stefan Edelkamp. Symbolic pattern databases in heuristic search planning. In M. Ghallab, J. Hertzberg, and P. Traverso, editors, *Proceedings of the 6th International Conference on Artificial Intelligence Planning and Scheduling (AIPS'02)*, pages 274–283, Toulouse, France, April 2002. Morgan Kaufmann.

Patrick Eyerich and Malte Helmert. Stronger abstraction heuristics through perimeter search. In Daniel Borrajo, Simone Fratini, Subbarao Kambhampati, and Angelo Oddi, editors, *Proceedings of the 23rd International Conference on Automated Planning and Scheduling (ICAPS'13)*, pages 303–307, Rome, Italy, 2013. AAAI Press.

Ariel Felner and Nir Ofek. Combining perimeter search and pattern database abstractions. In Ian Miguel and Wheeler Ruml, editors, *Proceedings of the 7th International Symposium on Abstraction, Reformulation, and Approximation (SARA-07)*, volume 4612 of *Lecture Notes in Computer Science*, pages 155–168, Whistler, Canada, 2007. Springer-Verlag.

Patrik Haslum, Adi Botea, Malte Helmert, Blai Bonet, and Sven Koenig. Domain-independent construction of pattern database heuristics for cost-optimal planning. In Adele Howe and Robert C. Holte, editors, *Proceedings of the 22nd National Conference of the American Association for Artificial Intelligence (AAAI'07)*, pages 1007–1012, Vancouver, BC, Canada, July 2007. AAAI Press.

Malte Helmert, Patrik Haslum, and Jörg Hoffmann. Flexible abstraction heuristics for optimal sequential planning. In Mark Boddy, Maria Fox, and Sylvie Thiebaux, editors, *Proceedings of the 17th International Conference on Automated Planning and Scheduling (ICAPS'07)*, pages 176–183, Providence, Rhode Island, USA, 2007. Morgan Kaufmann.

Malte Helmert, Patrik Haslum, Jörg Hoffmann, and Raz Nissim. Merge & shrink abstraction: A method for generating lower bounds in factored state spaces. *Journal of the Association for Computing Machinery*, 61(3), 2014.

Malte Helmert. The Fast Downward planning system. *Journal of Artificial Intelligence Research*, 26:191–246, 2006.

Peter Kissmann and Stefan Edelkamp. Improving cost-optimal domain-independent symbolic planning. In Wolfram Burgard and Dan Roth, editors, *Proceedings of the 25th National Conference of the American Association for Artificial Intelligence (AAAI'11)*, pages 992–997, San Francisco, CA, USA, July 2011. AAAI Press.

Peter Kissmann. *Symbolic Search in Planning and General Game Playing*. PhD thesis, Universität Bremen, 2012.

Kenneth L. McMillan. *Symbolic Model Checking*. Kluwer Academic Publishers, 1993.

Raz Nissim, Jörg Hoffmann, and Malte Helmert. Computing perfect heuristics in polynomial time: On bisimulation and merge-and-shrink abstraction in optimal planning. In Toby Walsh, editor, *Proceedings of the 22nd International Joint Conference on Artificial Intelligence (IJCAI'11)*, pages 1983–1990. AAAI Press/IJCAI, 2011.

Silvan Sievers, Martin Wehrle, and Malte Helmert. Generalized label reduction for merge-and-shrink heuristics. In Carla E. Brodley and Peter Stone, editors, *Proceedings of the 28th AAAI Conference on Artificial Intelligence (AAAI'14)*, pages 2358–2366, Austin, Texas, USA, January 2014. AAAI Press.

Álvaro Torralba and Vidal Alcázar. Constrained symbolic search: On mutexes, BDD minimization and more. In Malte Helmert and Gabriele Röger, editors, *Proceedings of the 6th Annual Symposium on Combinatorial Search (SOCS'13)*, pages 175–183. AAAI Press, 2013.

Álvaro Torralba and Jörg Hoffmann. Simulation-based admissible dominance pruning. In Qiang Yang, editor, *Proceedings of the 24th International Joint Conference on Artificial Intelligence (IJCAI'15)*, pages 1689–1695. AAAI Press/IJCAI, 2015.

Álvaro Torralba and Peter Kissmann. Focusing on what really matters: Irrelevance pruning in merge-and-shrink. In Levi Lelis and Roni Stern, editors, *Proceedings of the 8th Annual Symposium on Combinatorial Search (SOCS'15)*, pages 122–130. AAAI Press, 2015.

Álvaro Torralba, Stefan Edelkamp, and Peter Kissmann. Transition trees for cost-optimal symbolic planning. In Daniel Borrajo, Simone Fratini, Subbarao Kambhampati, and Angelo Oddi, editors, *Proceedings of the 23rd International Conference on Automated Planning and Scheduling (ICAPS'13)*, pages 206–214, Rome, Italy, 2013. AAAI Press.

Álvaro Torralba, Carlos Linares López, and Daniel Borrajo. Symbolic merge-and-shrink for cost-optimal planning. In Francesca Rossi, editor, *Proceedings of the 23rd International Joint Conference on Artificial Intelligence (IJCAI'13)*, pages 2394–2400. AAAI Press/IJCAI, 2013.

Álvaro Torralba, Carlos Linares López, and Daniel Borrajo. Abstraction heuristics for symbolic bidirectional search. In Subbarao Kambhampati, editor, *Proceedings of the 25th International Joint Conference on Artificial Intelligence (IJCAI'16)*. AAAI Press/IJCAI, 2016.

Álvaro Torralba. *Symbolic Search and Abstraction Heuristics for Cost-Optimal Planning*. PhD thesis, Universidad Carlos III de Madrid, 2015.