Graph Pruning and Symmetry Breaking On Grid Maps

Daniel Harabor
NICTA and The Australian National University
Email: firstname.lastname@nicta.com.au

1 Introduction

Pathfinding systems that operate on uniform-cost grid maps are common in the AI literature and application areas such as robotics and real-time video games. Typical speed-up enhancements in such contexts include reducing the size of the search space using abstraction [Botea *et al.*, 2004] and developing new heuristics to more accurately guide search toward the goal [Sturtevant *et al.*, 2009]. Though effective each of these strategies has shortcomings. For example, abstraction methods usually trade optimality for speed. Meanwhile, improved heuristics usually require significant extra memory.

My research proposes to speed up grid-based pathfinding by identifying and eliminating symmetric path segments from the search space. Two paths are said to be symmetric if they are identical save for the order in which the individual moves (or steps) occur. To deal with path symmetries I decompose an arbitrary grid map into a set of empty rectangles and remove from each rectangle all interior nodes and possibly some from along the perimeter. A series of macro edges are then added between selected pairs of remaining nodes in order to facilitate provably optimal traversal through each rectangle. The new algorithm, Rectangular Symmetry Reduction (RSR), can speed up A* search by up to 38 times on a range of uniform cost maps taken from the literature. In addition to being fast and optimal, RSR requires no significant extra memory and is largely orthogonal all existing speedup techniques. When compared to the state of the art, RSR often shows significant improvement across a range of benchmarks.

2 Related Work

Symmetry is often an undesirable characteristic in a search space. In the presence of symmetry, search algorithms can evaluate many equivalent states and make little real progress toward the goal. The problem of how best to deal with symmetry has received significant attention in areas such as planning [Fox and Long, 1999] and constraint programming [Gent and Smith, 2000] but there are very few works that explicitly identify and deal with symmetry in pathfinding domains such as grid maps.

Pochter *et al.* [2010] describe a search space reduction algorithm which bears some similarity to RSR. Their approach involves decomposing a map into so called *Swamp* areas that can be ignored because there always exists a symmetric path that does not involve any nodes from the swamp. Each search instance is then limited to a corridor of provably necessary swamp and non-swamp areas. All remaining nodes in the graph are ignored. The focus of this approach is therefore on identifying regions of the search space that can be ignored. By comparison, RSR tries to reduce the search effort involved in exploring any given area. Thus the two methods have complementary aims.

A similar graph pruning technique, known as as the *dead-end heuristic*, is due to Björnsson and Halldórsson [2006]. Like RSR, this method decomposes grid maps into obstacle-free zones connected by entrances and exits. A preliminary online search in the decomposed graph is then used to identify zones that are not relevant for reaching the goal. The central idea is to avoid exploring such "dead ends" during a subsequent search in the original grid. This approach is similar to the one employed by Pochter *et al.* [2010] but reported results suggest it is not as effective.

3 Symmetries in 4-connected Grid Maps

I have previously looked at the problem of symmetry breaking in 4-connected grid maps, which allow straight but not diagonal movement. Although less popular than 8-connected grid maps, this domain appears regularly in the literature [Yap, 2002] and is often found in the pathfinding systems of modern video games; e.g Square Enix's *Children of Mana* (Gameboy Advance) and Astraware's *My Little Tank* (iPhone).

In [Harabor and Botea, 2010] my co-author and I propose the following offline strategy for identifying and eliminating symmetric paths in 4-connected grid maps:

1. Decompose the grid map into a set of empty rooms, where each empty room is rectangular in shape and free of any obstacles. The size of the rooms can vary across a map, depending on the placement of the obstacles.
2. Prune all nodes from the interior but not the perimeter of each empty room.
3. Add a series of *macro edges* that connect each node on the perimeter of an empty room with a node on the directly opposite side of the room [1]. The cost of each edge

[1] Alternatively, macro edges could be generated on-the-fly during search. This obviates the need to store them explicitly.

is equal to the Manhattan distance between its two endpoints.

To handle cases where the start or goal location is a node previously pruned, we use an insertion procedure that re-adds single nodes back into the graph for the duration of the search. Each insertion operation can be performed in constant time. Furthermore, if the start and goal are located in the same empty rectangle an optimal path is immediately available – obviating the need for search.

We prove the optimality of this approach and evaluate it empirically by comparing the performance of A* when searching on pruned vs. un-pruned grids. We run experiments on a wide range of realistic game maps including one well known set from the game *Baldur's Gate II*. Results show that in many cases over 50% of all nodes on a given map can be pruned, resulting in improvements to average search times by a factor of up to 3.5.

4 Symmetries in 8-connected Grid Maps

I am currently studying symmetry breaking methods for the much more common 8-connected grid map domain. To that end, my co-authors and I propose Rectangular Symmetry Reduction (RSR): a new algorithm which extends the empty rooms decomposition described in [Harabor and Botea, 2010] in several directions: (i) we generalise the method from 4-connected grid maps to the 8-connected case where the branching factor makes effective symmetry elimination more challenging; (ii) we develop a stronger offline pruning technique to further reduce the number of nodes which need to be explored during search; (iii) we give a novel online pruning strategy which speeds up node expansion by selectively evaluating either all neighbours associated with a particular node or only a small subset. We then prove that each proposed extension preserves both optimality and completeness during search.

We evaluate RSR on a range of synthetic and realistic benchmarks from the literature and find that it can improve the average performance of A* by up to 38 times. Compared to Harabor and Botea's method [2010], we both extend the applicability and improve the speed on the subset of instances where both methods are applicable. When compared to the Swamp-based method of Pochter *et al.* [2010], we find that the two algorithms have complementary strengths and identify classes of instances where either RSR or Swamps is more suitable. We find that Swamps are better on maps with small open areas and their effectiveness reduces on maps with larger open areas. The opposite is true for RSR: larger open areas allow for larger empty rectangles, leading to a corresponding improvement in performance. Our results to date are encouraging: we identify a wide range of instances where RSR is clearly the better choice, dominating convincingly across a range of benchmarks.

5 Future Work

There are several directions available for future work; I plan to pursue each of these during the remainder of my doctoral candidature:

- **Alternative decomposition methods:** In the presence of large open areas, RSR can often compute optimal paths much faster than searching on the original map. On less favourable map topographies we achieve more modest improvements. I would therefore like to explore alternative decomposition techniques, based for example on convex shapes, which would allow bigger empty regions to be identified and lead to better performance.
- **Stronger online pruning techniques:** RSR can produce map decompositions in which individual nodes have a high branching factor. This is undesirable as considering a large number of neighbours usually slows down search. I intend to address this problem through the development of stronger pruning techniques that can be applied during the online phase of a pathfinding search.
- **Synthesis with existing pruning methods:** RSR is orthogonal to almost all existing techniques for speeding up pathfinding. It could therefore be integrated as part of a larger framework involving specialised heuristics or other graph pruning and state space reduction techniques; for example as described in [Botea *et al.*, 2004; Björnsson and Halldórsson, 2006; Pochter *et al.*, 2010].
- **Generalisations to other domains:** I am currently investigating whether breaking symmetric paths can be applied to more general domains. For example: weighted grids or road networks. The former often appear in application areas such as robotics and video games while the latter are particularly important in GPS navigation.

References

[Björnsson and Halldórsson, 2006] Y. Björnsson and K. Halldórsson. Improved heuristics for optimal pathfinding on game maps. In *AIIDE*, pages 9–14, 2006.

[Botea *et al.*, 2004] A. Botea, M. Müller, and J. Schaeffer. Near optimal hierarchical path-finding. *J. Game Dev.*, 1(1):7–28, 2004.

[Fox and Long, 1999] M. Fox and D. Long. The detection and exploitation of symmetry in planning problems. In *IJCAI*, pages 956–961, 1999.

[Gent and Smith, 2000] I. P. Gent and B. M. Smith. Symmetry breaking in constraint programming. In *ECAI*, pages 599–603, 2000.

[Harabor and Botea, 2010] D. Harabor and A. Botea. Breaking path symmetries in 4-connected grid maps. In *AIIDE*, pages 33–38, 2010.

[Pochter *et al.*, 2010] N. Pochter, A. Zohar, J. S. Rosenschein, and A. Felner. Search space reduction using swamp hierarchies. In *AAAI*, 2010.

[Sturtevant *et al.*, 2009] N. R. Sturtevant, A. Felner, M. Barrer, J. Schaeffer, and N. Burch. Memory-based heuristics for explicit state spaces. In *IJCAI*, pages 609–614, 2009.

[Yap, 2002] P. Yap. Grid-based path-finding. In *Canadian AI*, pages 44–55, 2002.