Benchmarks for Grid-Based Pathfinding

Nathan R. Sturtevant

***Abstract*—The study of algorithms on grids has been widespread in a number of research areas. Grids are easy to implement and offer fast memory access. Because of their simplicity, they are used even in commercial video games. But, the evaluation of work on grids has been inconsistent between different papers. Many research papers use different problem sets, making it difficult to compare results between papers. Furthermore, the performance characteristics of each test set are not necessarily obvious. This has motivated the creation of a standard test set of maps and problems on the maps that are open for all researchers to use. In addition to creating these sets, we use a variety of metrics to analyze the properties of the test sets. The goal is that these test sets will be useful to many researchers, making experimental results more comparable across papers, and improving the quality of research on grid-based domains.**

***Index Terms*—path planning, pathfinding, grid, search, map**

I. Introduction

Grid-based maps have been widely used as test domains in a number of fields, but until recently there was no publicly available standardized repository of problems for researchers. As a result we have created a pathfinding repository (http://www.movingai.com/benchmarks/) for storing both maps and problem sets that can be run on the maps.

This paper describes the data sets currently available online. We do not set out to describe a particular scientific advancement here. Instead, the goal is to describe the repository now available, with the intent that the distribution of these maps and problem sets will improve the quality and comparison of work that uses grids as test domains.

We encourage researchers to test algorithms on as many maps as possible from the test sets, as this will help show the generality or the limitations of a given approach. Additionally, if researchers are working on techniques for video games, they will find several sets of maps taken from commercial games.

In short, our goal is that this will improve the quality of evaluation in our own and others' scientific work.

We also describe a number of metrics which help typify the characteristics of each map. These help distinguish the features of each map type which might influence performance or the applicability of a given approach.

When working on these test sets, we initially formed a number of hypothesis about the properties of these maps. We state them here with some justification, and then analyze how strongly these predictions hold in practice.

- Scaling maps does not preserve all of the underlying properties of a map. Scaling a map will create larger open spaces and larger openings between distinct areas in the original map. This isn't to say that it is 'wrong' to scale a map, but that this change should be understood.
- Artificial maps created algorithmically have different properties than maps created by designers for particular applications. If an algorithm is being designed for a particular application, it is important that the maps being tested are similar to the application under consideration.
- There is a significant difference in the properties of a game map, depending on what genre of game the map was designed for. In role-playing games, players tend to have a linear experience through a map, while real-time strategy games are more about the strategic use of space, leading to less linear maps.

The rest of the paper is as follows. We begin by describing the map format, as well as the types of maps available in the repository, and the source of these maps. Then we describe how the problem sets were built and the metrics that we use for measuring the properties of the maps. Finally, we show a number of experimental results across the domains and provide a rough classification of the maps. We conclude with suggestions on use of these metrics.

N. Sturtevant is with the Department of Computer Science, University of Denver, Denver CO, 80208 USA e-mail: sturtevant@cs.du.edu.

II. Map Types

Each of the following maps is available from an online pathfinding repository at http://movingai.com/benchmarks/. They are also available for anonymous SVN checkout from a google code repository:

```
svn checkout http://hog2.googlecode.com/svn
/trunk/maps/ hog2-maps
```

All maps online are stored in the following format; some maps in the repository use a more complex representation.

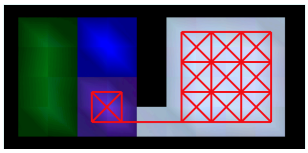
A. Map File Format Description

The map file format used in the repository was originally developed by Yngvi Björnsson and Markus Enzenberger at the University of Alberta. We adapted this format as we began importing different types of maps from commercial games, but only the simplest format is described here.

The map format begins with the type, which is always 'octile', followed by the dimensions of the map and then the map data itself. All cells in a map are either blocked or unblocked. However, to represent the underlying maps in a more artistic manner, several types of terrain have been added. Normal ground is represented by a period ('.'), and shallow water is represented by the 'S' character. These are the only passable types of terrain. All other terrain is considered to be impassable, including trees ('T'), water ('W') and out of bounds ('@'). This is illustrated in Figure 1. Part (a) of the figure shows the textual representation of the map, while part (b) shows the graphical representation. Overlaid on the map is the graph which represents the movement that can be taken on the map. In this case, an agent in the map can walk between

```
type octile
height 5
width 10
map
@@@@@@@@@@
TTWW@....@
TTWW@....@
TTSS@....@
TTSS.....@
```

(a)



(b)

Fig. 1. Sample map text (a) and graphical (b) representation. The lines show the connectivity between grid cells.

(a) (b)

Fig. 2. Sample map from Baldur's Gate II (a) and Dragon Age: Origins (b).

open ground and shallow water, but cannot traverse any other terrain.

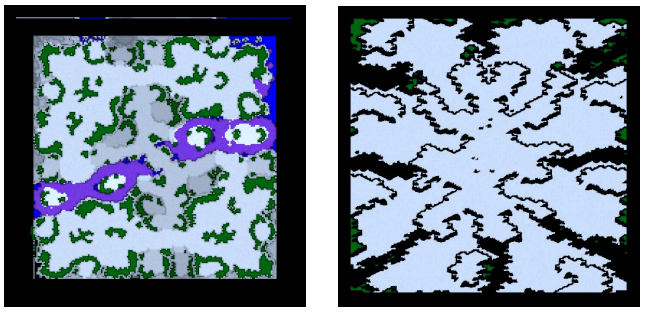
As shown in Figure 1, we assume that diagonal moves are only possible if the related cardinal moves are both possible. That is, it is not possible to take a diagonal move between two obstacles, and it is not possible to take a diagonal move to cut a corner. We make this assumption because in a real game all creatures occupy some volume of space and are not able to pass through blocked corners. The movement in the game Dragon Age: Origins, for instance, works with this assumption.

Following are the maps stored in the repository.

B. Baldur's Gate II

The set of 120 maps from Baldur's Gate II (BG) is one of the more extensively used map sets in previously published papers. BG is a Role-Playing Game (RPG) by BioWare Corp. published in 2000. The game uses grids internally for representing maps. These maps were originally extracted by Yngvi Björnsson when he was at the University of Alberta. The largest map in this set, with 51,586 passable states, is shown in Figure 2(a).

Sturtevant and Buro [1] scaled these maps to 512×512 and then generated test sets by randomly generating paths within the maps. Maps where it was difficult to generate paths of length 512 were thrown out, so the set used for experimentation only uses 75 maps, instead of the 120 unscaled maps in the repository.



(a) (b)

Fig. 3. Sample map from Warcraft III (a) and Starcraft (b).

C. Dragon Age: Origins

Dragon Age: Origins (DAO) is another RPG developed by BioWare Corp. and published in 2009. Like Baldur's Gate, DAO uses grids as the internal map format. An automated abstraction mechanism [2] speeds the pathfinding process. We extracted this maps with help from BioWare and our student James Balasalle. BioWare has explicitly given permission to distribute these files as benchmark problems.

The 156 maps represent the set of maps that shipped with the original game. This gives an example of the distribution of maps found in a commercial game, and allows comparisons across the full set of problems. The largest map has 137,375 walkable states, while some maps only have a few hundred walkable states. The map in Figure 2(b) has 96,603 states and paths of length up to 2800.

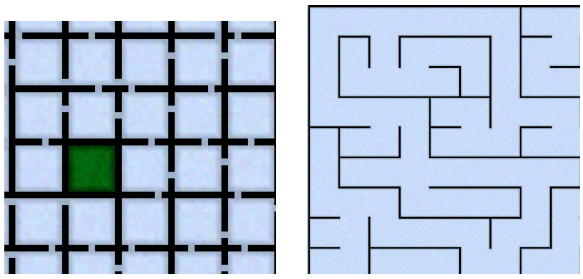
Note that the data in these maps is walkability data which is occasionally blocked by invisible physics objects in the world. So, there are some maps which contain more states than would actually be traversable in the game.

D. Warcraft III

Warcraft III (WCIII) is a Real-Time Strategy game (RTS) published in 2002 by Blizzard Corp. Although the map format in the game is based on grids, there is other meta-data, such as height information, which makes the representation more complicated. In the repository we have converted the format down to a simple grid-based format by adding borders around cells with height differences. There were originally 43 maps in this set, which were extracted directly from the game resources. This has been reduced to 36 maps which are topologically more interesting. The maps have been post-processed, removing all but the largest connected component from the map. The maps in the repository are scaled to 512×512. Unscaled maps in a different format are in the svn repository. A sample Warcraft III map can be found in Figure 3(a).

E. Starcraft

Starcraft (SC) is another RTS published in 1998 by Blizzard. It has maintained its popularity over the last decade primarily through competitive play. The repository contains 76 maps extracted from public internet archives and converted to the



(a) (b)

Fig. 4. Portions of sample room (a) and maze (b) maps.

standard format by Dave Churchill, a graduate student at the University of Alberta. The maps were then post-processed to remove all but the largest connected component (other areas have been marked as trees). All but a few maps are 512×512 or larger. A sample Starcraft map can be found in Figure 3(b).

It should be clear from Figures 2 and 3 that the RTS maps have different designs than the RPG maps.

F. Room Maps

We originally created the room maps for our work [3] as an alternative to existing map sets. The original intent was to create maps with larger local minima that could be scaled to larger sizes. The room maps are uniformly filled with square rooms of fixed size. Then, with a probability of 0.8, a random cell between two adjacent rooms is un-blocked. The original set of room maps was 8×8. The repository also contains rooms sizes 16×16, 32×32 and 64×64. This map set differs slightly from previous sets in three ways. First, the maps are 512×512 instead of 256×256. Next, all but the largest connected component were removed from the map. Finally, doors were adjusted to not appear on the corners between rooms. The repository contains 10 maps for each room size. A portion of an example room map can be found in Figure 4(a). In this map there is one room which had no openings, and thus is blocked.

Note that a map with 16×16 rooms is not identical to a map with 8×8 rooms that has been scaled up by a factor of two. In particular, in the 16×16 map the walls will be thinner and the passages between rooms will be smaller (1 cell instead of 2 cells).

G. Mazes

Mazes are another type of artificial map. The mazes are formed randomly, with a number of parameters influencing the layout of the map. Initially the whole map is blocked except for a single square in the center. The algorithm attempts to extend the current maze corridor, but has a (3%) chance of randomly selecting a different corridor to extend. All maps in this set are 512×512, and are parameterized by the corridor size in the passages, which can be 1, 2, 4, 8, 16, or 32. A portion of an example maze with corridor size of 16 can be found in Figure 4(b).

Similar to the room maps, a scaled maze would have thicker walls than a map created with a larger corridor size, which always has walls that are 1 cell thick.

H. Random maps

Random maps are created by randomly blocking grid cells within a map. The number of blocked cells in each set of maps varies between 10-40% by 5% increments. Similar to previous maps, only the largest connected component is left in the map. Maps which are 45% or more blocked have only very small connected components and are not interesting. We block exactly the given percentage of states in any map. The number of walkable states in a map varies, however, because unreachable areas are blocked with trees.

III. Problem Set Generation

Most problem sets for these maps were selected by randomly picking two points in the world and then finding the optimal path between them. The optimal path length is measured with diagonals having cost $\sqrt{2}$, and cardinal movement having cost 1. Given the optimal solution length, the problems were categorized into larger 'buckets'. Reporting results in buckets reduces the variance and makes it easier to compare to similar problems across maps. The bucket for a path of length ℓ is $\lfloor \ell/4 \rfloor$. Each bucket on each map contains at most 10 problems. The problem sets were built by choosing 100,000 pairs of random points. The largest contiguous set of full buckets was retained in the problem set, while unfilled buckets were discarded.

One exception to this is the BG set, which fixed the maximum path length at 512, corresponding to the 127th bucket. A few maps in this set do not have 10 problems in the 127th bucket. We maintain this set as originally built for historical purposes in order to match published results with these maps.

The maze sets with corridor width 1 were built with 200,000 sample points, as 100,000 points was insufficient to fill the buckets with problems.

IV. Map Metrics

Given the maps defined in the previous section, we use a number of metrics to characterize each map set, described here. Full metrics, including per-map results, can be found on the benchmark web site. These metrics are meant to be easy to compute, so we use approximate measures where a full computation would have significant overhead.

A. Number of states

The first metric is simple: the number of unblocked tiles in the map. Each set of artificially generated maps generally has the same number of states, however in maps created for a real game, there can be significant variance in map sizes. In the DAO set the largest maps have over 100,000 states, while the smallest just have a few hundred.

B. Estimated maximum length shortest path

Finding the exact longest path in each map via an all-pairs shortest-path computation is expensive. With over 100,000 states in a map, the all-pairs shortest-path data would require $10^{10}/2$ entries. There are cheaper ways to find the longest

path, but a reasonably accurate estimation is sufficient for our purposes.

For each map we choose a random point and then perform a Dijkstra search until the last state is found. Then, we measure the longest path from this state. We do this 5 times for each map and take the maximum. Results are then averaged over all maps in a set. Mazes have, by far, the longest paths. We will discuss other correlations later.

C. Transit Node Count

The notions of transit nodes [4] and highway dimension [5] are metrics that have been used to describe why recent planning approaches for road networks have been so successful. Transit nodes in a graph are the nodes at some small radius which are on all shortest paths to nodes at a larger radius. When the number of transit nodes in a road graph is sparse, the all-pairs shortest-path data can be stored just between transit nodes, greatly reducing the storage overhead.

This concept has been generalized into the concept of highway dimension. Abraham et. al. [5] show that low highway dimension guarantees efficient pre-processing and shortest path queries for a number of planning algorithms, including contraction hierarchies [6] and reach [7].

Highway dimension is difficult to compute directly, because it requires an expensive optimization. Instead, we measure the number of transit nodes at radius r given shortest paths to $4r$, for several values of r. This is a lower bound on the highway dimension, and is measuring the same property – whether a sparse set of points can be found on all shortest paths.

We randomly sampled 50 points on each map, measuring the number of transit nodes for each point, and then average the results over all maps in each set. We report results for $r = 10$ and $r = 40$. If a map was completely explored before reaching depth $4r$ the results were excluded from the reported averages.

A low highway dimension and/or transit node count suggests that memory-efficient heuristics can be built to solve pathfinding problems quickly.

D. Dimension

In addition to transit node count, we attempt to directly estimate the dimension of a map. This is related to how the branching factor might be estimated in an exponential tree. We do this by choosing a random point in a map and then performing a Dijkstra search, recording the number of nodes expanded at each depth. Then we fit the number of states at each depth to the polynomial $a + b \cdot x + c \cdot x^2$ using linear regression. (The whole process is also called quadratic or polynomial regression.) The value for c gives some indication of the dimension of the map. If $c = 0$, the map is one dimensional. If $c = 1$ the map is two dimensional. If $c < 0$, the number of states at each new depth is being reduced. The number of nodes expanded at the end of the search does tend to tail off as the ends of the map are reached, so we only perform the regression on the node counts from the first half of the search. We repeated this measurement 50 times on each map. Results are then averaged over all maps.

Originally, our dimension metric was intended to estimate the dimensions of a Euclidean heuristic [8], but experiments showed that results from these two approaches did not correlate.

E. Heuristic Quality and Nodes Expanded

We use the problem sets to measure the quality of the search heuristic in two ways. For all problems in bucket 127, which have length 508-512, we compare the initial heuristic value to the optimal path cost and report the percentage. Many small maps do not have paths of this length, so this is biased towards larger maps. We also measured the number of nodes expanded by A* when solving these problems and report this as well.

V. Experimental Results

The average metrics for each problem set are in Table I. The first column is the map set. Results for the BG and WCIII map sets are provided both for the original sizes and scaled to 512×512. The artificial maps are all 512×512, while the other maps are at their native resolutions.

The reported metrics are averages of the number of states in the map, the maximum length shortest path, the dimensionality, the transit node count (TNC), and, for problems with optimal length 508-512, the heuristic accuracy and nodes expanded.

As the values reported are averages, there can be significant variation, especially in the DAO set which contains a number of small maps. We will look at this broad classification of values; detailed results are available online.

The following data points are worth noting in the set; further analysis follows in the next section.

- Increasing the size of the room maps increases the number of states in the map (because there are fewer walls) and decreases the heuristic accuracy. The number of nodes expanded in paths length 508-512 increases slightly as the rooms get larger.
- Increasing the size of the maze corridors also increases the number of states in the map. It also increases the heuristic accuracy and the number of nodes expanded. Node expansions increase because larger corridors provide more nodes for A* to expand. Heuristic accuracy increases because the maximum path length decreases.
- As the random maps are progressively more filled, the number of states in the map decreases, but the nodes expanded increases. This is mirrored by a drop in heuristic accuracy, meaning that the problems get harder as more random obstacles are added. But, when the map is 40% filled with obstacles, the problems begin to get easier again.
- The transit node count is significantly higher at $r = 10$ than at $r = 40$ on the room-64 and maze-32 maps. Recall that the transit node count is measured relative to distances r and $4r$. At radius 40, most states will still be inside a room of size 64, increasing the TNC significantly. But, once the bulk of a search exits a door to a room, all optimal paths will go through the same door, decreasing

Map Set	Average Num. States	Average Max Path	Dimension	Tran. Node Cnt. radius = 10	Tran. Node Cnt. radius = 40	Heuristic Accuracy (512)	Expanded Nodes (512)
BG	4,507	150.5	0.52	6.12	2.04	-	-
BG512	73,930	693.0	0.44	39.67	31.74	0.76	22,375.0
DAO	21,323	427.4	0.31	11.05	3.32	0.59	17,857.6
SC	263,782	1,190.2	0.41	40.11	15.44	0.78	26,993.3
WCIII	10,669	155.5	0.70	6.26	2.06	-	-
WCIII512	90,910	674.0	0.75	32.27	13.30	0.84	27,386.4
Rooms-8	206,792	891.7	0.78	5.32	9.15	0.86	40,414.8
Rooms-16	231,263	863.8	0.94	3.94	6.54	0.83	41,776.3
Rooms-32	243,733	881.0	0.97	4.16	4.77	0.83	45,210.7
Rooms-64	249,352	931.0	0.81	30.04	3.20	0.77	50,467.7
Mazes-1	131,071	7,575.6	0.01	1.82	1.82	0.18	4,619.0
Mazes-2	174,517	5,795.1	0.02	1.86	1.73	0.24	8,671.6
Mazes-4	209,268	4,849.1	0.03	1.90	1.81	0.27	14,972.9
Mazes-8	232,928	4,526.0	0.03	2.14	1.85	0.34	19,768.0
Mazes-16	246,042	3,826.5	0.02	5.81	1.93	0.41	27,693.8
Mazes-32	253,819	2,666.4	0.03	22.89	2.77	0.49	39,123.8
Random-10	235,903	766.8	1.13	16.93	22.79	0.99	14,696.4
Random-15	222,689	789.9	1.01	14.34	19.97	0.97	21,728.9
Random-20	209,255	816.6	0.92	11.70	16.80	0.95	28,408.8
Random-25	195,315	851.1	0.79	9.33	13.68	0.91	33,447.2
Random-30	180,209	894.4	0.64	6.81	10.81	0.86	34,956.9
Random-35	161,313	983.5	0.50	4.26	6.86	0.76	35,540.7
Random-40	96,365	1,585.1	0.09	2.42	2.36	0.46	19,555.8

TABLE I
AVERAGE RESULTS FOR EACH MAP SET.

the TNC. A similar effect occurs in mazes with larger corridor sizes.

- The maze maps are the only set with dimension of approximately 0. As the search is constrained by the maze corridors, no matter the size of the corridor, the number of new states will tend to grow linearly with the depth.

Next we evaluate our three original hypothesis.

A. Evaluating Map Scaling

The first question we look at is the effect of scaling. We look primarily at the BG maps, but the same trends hold for the WCIII maps.

The unscaled maps have an average of 4,507 states, versus 73,930 in the scaled maps. This is, of course, the motivation for initially scaling these maps, as the originally maps were quite small and didn't represent the work that might be required in a modern game. Our original hypothesis was that some properties of maps are changed by scaling.

Results show that our measure of dimension is relatively unchanged by scaling, while TNC is significantly changed by scaling. We plotted the dimension of the unscaled versus scaled maps, and then fit the resulting point cloud to a line. The points fit the line $y = 0.01 + 0.98x$ with a correlation coefficient of 0.96. This confirms that scaling a map leaves the dimensionality unchanged. Our intuition behind this is that the relative size of areas stay the same when a map is scaled, so although more nodes will be expanded, the relative ratios of nodes expanded at each depth stays the same.

Transit node count, however, changes significantly. The unscaled BG maps have 6.12 transit nodes at radius 10 and 2.04 transit nodes at radius 40. This is a significant departure from the result for the scaled maps, which have 39.67 transit nodes at radius 10 and 31.74 transit nodes at radius 40. Given normal tie-breaking rules (for expanding states with highest g-cost first) on an empty map all states at radius r will be on the shortest path to states at radius $4r$. Scaling maps creates larger open areas, and thus the number of transit nodes increases. Note, however, that this measurement depends on the radius. Given measurements at larger radii, we would expect the number of transit nodes to decrease.

These results suggest that scaling is not necessarily the wrong thing to do – it depends on how the maps are used. But the change in transit node count suggests that heuristics could be less effective on scaled maps than unscaled ones.

B. Algorithmic Versus Designed Maps

Next we look at the maps which were generated algorithmically in comparison to the other map sets. We performed analysis of the metrics both manually and using k-means clustering. All metrics were normalized (i.e. linearly mapped to [0, 1]), and then k-means clustering was run 1000 times with 4 and 7 clusters of maps. We kept track of how many times any pair of maps was placed in the same cluster with each approach, and then looked at the top 20 pairings.

As expected, the artificial benchmark problems were very commonly grouped together, as the metrics for these maps are very similar. The exceptions to this was the Rooms-64 set, which is closest to the SC map set, and the Random-40 which is closest to the DAO map set. With 40% random obstacles the optimal paths in a map no longer resemble a straight line, as they do with fewer obstacles. Instead the obstacles form areas more analogous to the rooms and corridors which are seen in the DAO maps. The BG512 and WCIII512 map sets were almost always grouped together and show reasonable

similarity. This shows that in some cases algorithmically designed maps share metrics with human-designed maps, but not in the majority of cases.

C. Genre-Specific Differences

Finally, we look at the difference between maps for role-playing games (RPGs) and real-time strategy games (RTS). We focus on the DAO and SC sets which are not scaled. We see that the SC maps have more states and longer paths on average. They also have much higher TNC. This is partially a selection bias – the SC maps are being used in competitions and are not the maps that would be used in a play-through of a game. But, on the SC maps, a typical problem of length 512 would require 10% of the map states to be expanded, while the DAO problems would require 50% of the map states to be expanded. This is computed by looking at the average number of states in maps with paths length 508-512. This suggests that the SC maps are more two-dimensional: A heuristic search across a two-dimensional plane will expand a smaller fraction of the total states than a heuristic search along a one-dimensional line. But, this is contradicted by our measure of dimension, which shows the SC maps with 0.41 versus the DAO maps with 0.31. Further analysis revealed the difference: the average dimension of maps that contain paths of length 400 or longer is 0.07. Thus, larger DAO maps have lower dimension. This reveals that, for larger maps, there is a clear structural difference between SC and DAO maps.

VI. Conclusions

This paper describes a new repository that has been placed online to improve the evaluation of grid-based problems. This repository will allow researchers to use the same problems and test sets, increasing the reproducibility of published results. We introduce the data sets, describe how they were built, and provide metrics for distinguishing map types.

We provide the following suggestions for researchers using these problem sets:

- Report exactly which map set was used.
- Report exactly what problems were used. For example, a given bucket size across all maps in a particular problem set.
- Use the existing problem sets. This will allow other researchers to exactly duplicate your results.
- Report any deviation from the testing conditions. These could include different edge costs (e.g., 1.5 for diagonals for more efficient search), a different heuristic, or allowing agents to cut corners.
- Use a broad set of test maps to clearly illustrate the strengths and weaknesses of a given approach. For instance, techniques that work particularly well on maps with axis-aligned obstacles should not be exclusively tested on artificial benchmarks. If you are unsure of the limitations of your approach, use the benchmarks to explore them, and test to see if any of the provided metrics correlate with performance.

We have made some effort to collect an interesting set of maps and provide useful metrics to analyze them. But, we are open to adding more map sets or metrics if they can be demonstrated to provide useful classification. We are also open to adding other map formats to the repository.

Authors are encouraged to send us any papers published using these benchmarks as we are maintaining a list of papers which use them. Currently over 30 published papers have used these maps or benchmark problems starting in 2004 [9] and continuing to the present [10]. We hope that many more researchers will benefit from their broader public distribution.

Acknowledgements

We acknowledge the help of Chris Rayner in helping to measure the correlation between our dimension metric and the dimension computed by Euclidean heuristics.

References

[1] N. R. Sturtevant and M. Buro, "Partial pathfinding using map abstraction and refinement," in AAAI, 2005, pp. 1392–1397.
[2] N. R. Sturtevant, "Memory-efficient abstractions for pathfinding," in AIIDE, 2007, pp. 31–36.
[3] N. R. Sturtevant, A. Felner, M. Barrer, J. Schaeffer, and N. Burch, "Memory-based heuristics for explicit state spaces," in IJCAI, 2009, pp. 609–614. [Online]. Available: http://ijcai.org/papers09/Papers/IJCAI07-107.pdf
[4] H. Bast, S. Funke, and D. Matijevic, "Transit ultrafast shortest-path queries with linear-time preprocessing," in In 9th DIMACS Implementation Challenge, 2006.
[5] I. Abraham, A. Fiat, A. V. Goldberg, and R. F. Werneck, "Highway dimension, shortest paths, and provably efficient algorithms," in Proceedings of the Twenty-First Annual ACM-SIAM Symposium on Discrete Algorithms, ser. SODA '10. Philadelphia, PA, USA: Society for Industrial and Applied Mathematics, 2010, pp. 782–793. [Online]. Available: http://portal.acm.org/citation.cfm?id=1873601.1873665
[6] R. Geisberger, P. Sanders, D. Schultes, and D. Delling, "Contraction hierarchies: Faster and simpler hierarchical routing in road networks," in WEA, 2008, pp. 319–333.
[7] A. V. Goldberg, H. Kaplan, and R. F. Werneck, "Reach for a*: Efficient point-to-point shortest path algorithms," in IN WORKSHOP ON ALGORITHM ENGINEERING AND EXPERIMENTS, 2006, pp. 129–143.
[8] D. C. Rayner, M. H. Bowling, and N. R. Sturtevant, "Euclidean heuristic optimization," in AAAI, 2011. [Online]. Available: http://www.aaai.org/ocs/index.php/AAAI/AAAI11/paper/view/3594
[9] A. Botea, M. Müller, and J. Schaeffer, "Near Optimal Hierarchical Path-Finding," Journal of Game Development, vol. 1, no. 1, pp. 7–28, 2004.
[10] C. Hernández and J. A. Baier, "Fast subgoaling for pathfinding via real-time search," in ICAPS, 2011.

Nathan Sturtevant is Assistant Professor in the Computer Science Department at the University of Denver. His scientific research focuses on Artificial Intelligence and search in single and multi-agent settings, including applications to real-time environments, such as games, and more competitive environments.