

Some Experimental Results of Applying Heuristic Search to Route Finding

John Pearson* and Hans W. Guesgen†

Computer Science Department, University of Auckland
Private Bag 92019, Auckland, New Zealand
{jpea001, hans}@cs.auckland.ac.nz

Abstract

Search is often used to find shortest routes within real road networks. Standard search algorithms are memory intensive and inefficient for stand alone in-car systems. In this paper, we evaluate a number of heuristic search strategies in terms of space and time complexity for the route finding problem.

Route Finding and Search

Much research in recent years has focussed on developing technologies to enable improvements in the efficiency of road transport systems. Programs geared toward this goal include the TravTek (Rillings & Krage 1992) and Advance (Catling 1992) programs in the United States and SOCRATES (Catling & Harris 1995) in Europe. With each of these programs emphasis is placed on providing traveler information including route finding capabilities. With current technology and infrastructure, many current and proposed systems are of the decentralized autonomous variety, with processing and route map information stored on a local CD (Collier & Weiland 1994). Thus with high hardware costs, it is advantageous to keep the memory, computation, and storage device access to a minimum while keeping the accuracy of computed routes high (Dillenburg & Nelson 1995). Heuristic search offers the potential to reduce time and memory use in route finding systems by incorporating knowledge of the properties of the problem space. We evaluate heuristics for route finding and a number of strategies with potential to increase efficiency over standard heuristic search.

Uniform Cost Heuristic Search

Standard uniform cost search approaches, such as Dijkstra's method (1959), suffer the fate of combinato-

*Data and computing resources kindly provided by the Spatial Analysis Facility, University of Auckland.

†Supported by the University of Auckland Research Committee under grant A18/XXXXX/62090/F3414065.

rial explosion; the time and space complexity for such methods are of the order $O(b^d)$, where b is the branching factor and d is the depth of the search tree (Korf 1996). The A* algorithm (Hart & Nilsson 1968) incorporates heuristic search into uniform cost search, finding optimal solutions yet decreasing the effective branching factor. A number of heuristics which exploit the geometric nature of roading networks are applicable to the route finding problem and can be incorporated into A*.

We evaluated the air distance and Manhattan distance heuristics (the L2 and L1 metrics, respectively) in the Auckland road network and compared the decrease in node expansions over an uninformed search. Figure 1 depicts the node expansions created by the three heuristics for a shortest path between a location represented by the large dot on the left to a destination given by the large dot on the right.

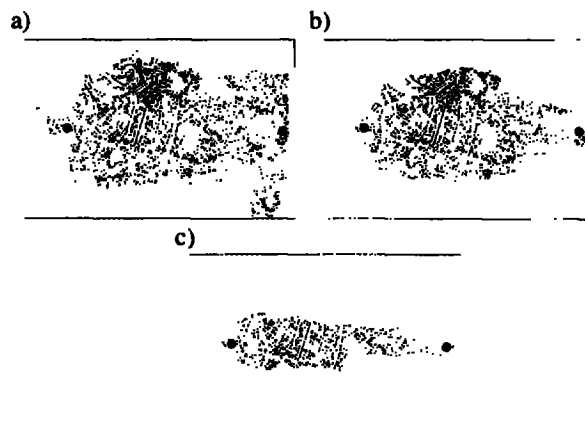


Figure 1: Node expansions for a) No heuristic, b) Air distance heuristic, c) Manhattan heuristic.

As can be seen from Figure 1, the uninformed brute force search expands almost every node in the search area in finding the goal state. With the air distance

heuristic the search is more focussed and the node expansions form a more elliptical path. The Manhattan heuristic is even more focussed with a decrease in the size of the minor axis of the ellipse over air distance. These results are consistent with Pearl's analysis of heuristics in a network of uniformly spaced and tightly connected cities (Pearl 1985). As the pruning power of the heuristic used increases, the pattern of expanded nodes reduces from a circular area to a more eccentric ellipse.

	<i>L2 Heuristic</i>	<i>L1 Heuristic</i>
Node Reduction	$\mu = 78.2\%$ $\sigma = 10.0\%$	$\mu = 83.3\%$ $\sigma = 13.4\%$
Path Inaccuracy	—	$\mu = 1.4\%$ $\sigma = 2.8\%$

Table 1: Results using L1 and L2 heuristics.

We evaluated the heuristics over 1000 randomly selected start-goal pairs, the results of which can be seen in Table 1. Both L1 and L2 heuristics produce a dramatic reduction in the number of node expansions with high average node reduction. Unlike the case for no heuristic or air distance, the Manhattan heuristic is inadmissible; that is it overestimates the actual cost to the goal and therefore cannot be guaranteed to produce optimal cost solutions. As can be seen from Table 1, the path inaccuracy rate of the Manhattan heuristic was only 1% above the optimal path length. Thus the more efficient Manhattan heuristic can be used in many situations where close to optimal solutions are acceptable.

Heuristic Depth First Search

A disadvantage of the uniform cost heuristic search methods is that while they reduce the number of node expansions, they only reduce the base of the order term for space complexity. This is of concern for low cost mobile route finding as memory is often limited. Korf's IDA* algorithm (1985), a heuristic depth first algorithm, is said to be asymptotically optimal in time and space in comparison to A*. It performs a series of bounded depth first searches to deliver an optimal solution while maintaining the space advantages of depth first search; space complexity of $O(d)$. However, this method is only useful for trees with integer-valued edge weightings, as control over the depth of iterations is lost.

Variants on IDA* for real-valued edges (Sarkar *et al.* 1991) in graphs together with cycle checking (Dillenburg & Nelson 1993; Taylor & Korf 1993) were implemented. These methods attempt to control re-expansion depth by estimating the total edge weight-

ing for the current iteration by looking at the expected number of nodes to be expanded. It was found that cycle checking increased computation time while varying edge branching factors in the road graph make controlled re-expansion impossible to manage (Pearson 1998). More appropriate methods for limited memory heuristic search maybe to use Chakrabarti *et al.*'s MA* algorithm (1989) or to use pruning to discard likely off-path nodes (Pearl 1985). Aside from these methods memory usage can be limited by introducing better heuristic strategies which limit both node expansions and memory usage.

Search Using Subgoals

As well as using an appropriate heuristic to guide search, use can be made of subgoals to reduce node expansions still further. Kuipers, as part of modeling human spatial cognition, suggests that human way finders use landmarks of previously traversed paths as part of route finding (Kuipers 1978). Korf also suggests that the merging of the planning concept of subgoals with search can improve search efficiency (Korf 1987). In particular, the complexity of such a search is dependent on only the most complex search between subgoals, $O(b^{d_{max}})$. Thus subgoal search methods not only act to decrease the effective branching factor but also the effective depth of a search.

Subgoal search methods have potential for route finding as knowledge of commonly traversed and obstacle avoiding intersections is often known prior to search. Dubois & Semet (1995) and Yang *et al.* (1991) propose algorithms which use topographical information to guide search around obstacles. Dillenburg & Nelson's Algorithm In (1995) incorporates knowledge of previously traversed nodes and thus topological knowledge is not required. These previously traversed nodes are potential subgoals and are referred to as multiple level island sets, *IS*. By using a parameter *E*, an estimate of the number of islands on the optimal path, the search can be guided through *E* islands out of the island set. Figure 2 shows islands for a search between two points, *S* and *G*, from Auckland city center to the suburb of St. Heliers. The islands *a* and *b* avoid the obstacle posed by Hobson Bay while island *c* is an intersection through which most traffic passes. For this situation an appropriate value for *E* would be 2.

We implemented Algorithm In and used Dillenburg & Nelson's suggested method for discovering islands. We split the Auckland road map into regions of 4 km² and calculated all shortest paths between the nodes of each pairs of regions, noting those nodes that appeared in 95% of all possible shortest paths between each set of region pairs. These nodes form a multiple

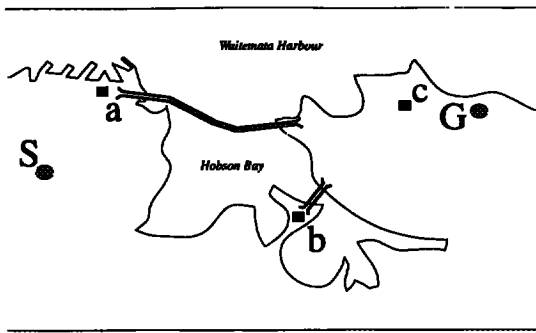


Figure 2: Islands for a search traversing Hobson Bay.

level island set for each set of region pairs. We found that for many regions we could not find nodes which reached the criteria of occurring in 95% of all routes. This is due in part to the density of the Auckland road network and a lack of real average travel times used in computing optimal routes. If real travel times could be used it is expected that the existence of islands would be more apparent.

Dillenburg & Nelson introduces a permuted heuristic to avoid island interference. Island interference can occur when the island heuristic guides search by taking into account a minimum path through a single island in isolation rather than through multiple islands. With a non-permuted heuristic a search in the situation in Figure 2 will guide the search directly to island *c*, when a path directed toward *a* then *c* or *b* then *c* would produce a more efficient search. We tested Algorithm In using a series of 700 randomly chosen start-goal pairs from two regions using air distance as a guiding metric. We found that using islands reduces node expansions by up to 10% over the non-island heuristic search. We also found that using a permuted heuristic reduces the number of node expansions over a non-permuted heuristic, especially when the value used for *E* overestimates the actual number of optimal lying islands (Pearson 1998). While the reduction in node expansions is not great, it is a significant improvement in efficiency. It would be beneficial to repeat these trials using actual travel times as we believe both efficiency and island discovery could be improved using realistic test data.

An Ordered Island Heuristic

The permuted heuristic requires $O(|IS|P_E)$ heuristic computations for each node generation. Thus as the expected number of optimal lying islands increases, the time taken to compute the heuristic can be greater than the saving made by a reduction in node expansions. Using branch and bound with this method re-

duces complexity but for larger numbers of islands computing the heuristic is still impractical. We found that excessive heuristic computation could be avoided by using an ordering of islands such that a combination of islands could be used in computing the heuristic. This reduced the complexity of heuristic computation to $O(|IS|C_E)$ while maintaining solution optimality.

Figure 3 shows a comparison of CPU times for the permuted, permuted branch and bound and ordered heuristics averaged over 700 trials. The figure indicates that the permuted and branch and bound heuristics require excessive amounts of CPU time, especially when the expected number of islands is large. Contrasting with this, the ordered heuristic requires considerably less CPU time with maximum CPU time occurring when the expected number of islands is half that of the size of the island set.

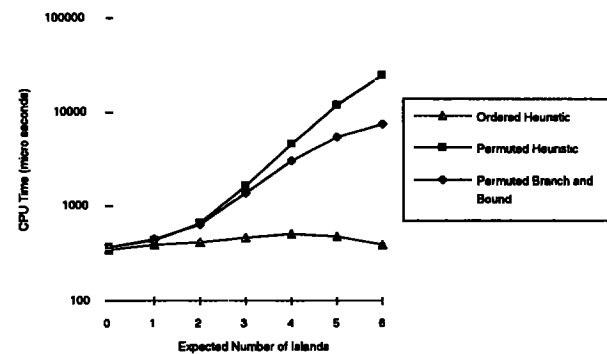


Figure 3: CPU Times for various island heuristics.

An ordering of potential subgoals is found by modifying the island discovery method. Each node found on the optimal paths between region pairs has its depth noted. When a choice is made for inclusion of a node in each island set the depth of the node is averaged over all routes that it occurs in. The average depth is then used to place nodes in the island set in order of increasing search depth.

Hierarchical Search

Hierarchical search methods can greatly reduce the complexity of search by simplifying the search into a series of searches among levels of smaller size. Each level in a hierarchical problem space is an abstraction of a lower level and thus a search through these levels has a lower complexity than a search in just a single base level. The use of hierarchies is well known in the AI literature but is also known in a number of other fields related to route finding. In the context of spatial reasoning Car & Frank (1994) and Timpf *et al.* (1992) explicitly recognize hierarchies as important as a model

for road navigation.

Korf (1987) discusses hierarchical search for route finding in the context of planning. He suggests two forms of abstraction for route finding: the subset model and the region model. The subset model is appropriate for route finding where total solution cost is important; nodes are preserved between levels and the simplification of the network is achieved by keeping and simplifying the more important edges at higher levels. A search then recursively proceeds through these abstraction levels in the following manner:

1. Search in the base state space to the nearest node in the next abstract space.
2. Search in the abstract space from the terminating node of the step 1 search to the nearest node in the abstract space to the goal node.
3. Search from the terminating node in step 2 to the goal node in the base state space.

The size of the graph that such a search must address is thus $O(\log N)$, where N is the size of the state space. Thus hierarchical search has potential in reducing search complexity for route finding. We implemented the hierarchical algorithm suggested by Car & Frank (1994) which uses the subset model. The algorithm uses a search space partitioned along the lines of road classifications. Level 0 represents the highest level of abstraction, level K being the base state containing all roads and intersections. By partitioning the space in this manner higher speed roads are placed at the highest level. Thus a search up through the hierarchy will not only simplify search but also maintain some level of path optimality.

Partitioning the search space creates regions formed by the boundaries of higher speed roads on the next hierarchical level. Thus a node at one level is related to the nearest node on the region boundary of the next highest level. The algorithm uses the hierarchical method as outlined above but will default to searching in just the base state space if the start and goal are deemed to be in the same region. The hierarchical network that we used for Auckland is depicted in Figure 4. Level 2 contains all roads, level 1 contains main roads and motorways, and level 0 contains just motorways.

Car & Frank's algorithm uses Dijkstra's method, but we use A* with the air distance heuristic to add an extra performance increase. Thus the method can be thought of as having two heuristic guiding principles: guidance using geometric distance together with the use of intermediate subgoals which guide the search through higher speed roads. Using the assumption

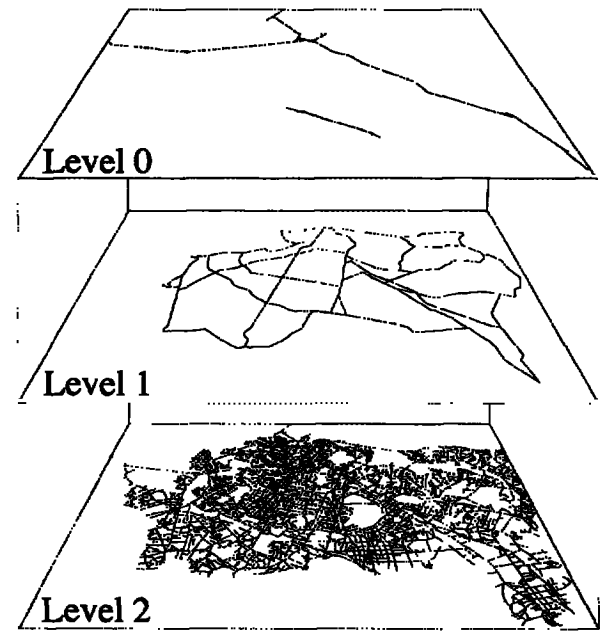


Figure 4: Hierarchies for the Auckland road network.

that optimal paths will always follow a hierarchical path means that the method is not guaranteed to be optimal, even though the air distance heuristic is admissible.

We conducted a series of trials using the hierarchical method and compared results with A* using the admissible air distance heuristic. We make a distinction between two types of result owing to the small size of the Auckland road map. Frequently a trial did not make use of the hierarchical network and defaulted to using A* in the base state space. Thus we present results which include searches in the lowest level (searches through all k to 0 levels) and results which include only those searches which make use of the hierarchical network (searches in levels $k - 1$ to 0).

	$K - 1$ to 0	K to 0
Node Decrease	81.7%	30.6%
Path Inaccuracy	13.1%	4.0%

Table 2: Results for 1000 trials of hierarchical A*.

As can be seen from the results in Table 2, for trials which include search at the lowest level both node reductions and path length increases are modest. Trials including just hierarchical search yield a large decrease in node expansions over flat A* but with a significant increase in path length over optimal. This path in-

crease maybe unacceptable for systems which place a high reliance on the accuracy of path length but maybe acceptable when weighed against the large reduction in search complexity. However, this result is due in large part to the construction of the hierarchical network used in testing. Regions were constructed at levels 1 and 0 which contained dead-end roads or leaf nodes. The algorithm does not always choose the most optimal route in these situations. Careful construction of the hierarchy of space spaces can limit these difficulties by including more edges at higher levels so that no leaf nodes exist.

Summary

This paper looks at the results of applying various heuristic search methods to route finding. We show that the Manhattan heuristic delivers more efficient solutions than the air distance heuristic for only a small increase in path cost. We implemented island search and found that search efficiency increased although we had difficulty in discovering islands. We examined the use of ordered island sets to decrease heuristic computation time and found this to be an effective method. Hierarchical search is an effective method of increasing search efficiency although we found that the path inaccuracy rate is very high. We hope to test these methods using real average travel times and assess their relative benefits in a more realistic context.

References

- Car, A., and Frank, A. 1994. General principles of hierarchical spatial reasoning - the case of wayfinding. In Waugh, T., and Healey, R., eds., *Advances in GIS Research, Proc. 6th International Symposium on Spatial Data Handling*, 10-12.
- Catling, I., and Harris, R. 1995. Socrates - from research to commercial implementation. In *Proc. Applications of Advanced Technologies in Transportation Engineering*, 588-593. New York, USA: ASCE.
- Catling, I., ed. 1992. *Advanced Technology for Road Transport IVHS and ATT*. Artech House. chapter 11 ADVANCE-The Illinois Dynamic Navigation and Route Guidance Demonstration Program, 247-270.
- Chakrabarti, P.; Ghose, S.; Acharya, A.; and de Sarkar, S. 1989. Heuristic search in restricted memory. *Artificial Intelligence* 1:197-221.
- Collier, W. C., and Weiland, R. J. 1994. Smart cars, smart highways. *IEEE Spectrum* 2(2):27-33.
- Dijkstra, E. 1959. A note on two problems in connexion with graphs. *Numerische Mathematik* 1.
- Dillenburg, J. F., and Nelson, P. C. 1993. Improving the efficiency of depth-first search by cycle elimination. *Information Processing Letters* 45(1):5-10.
- Dillenburg, J. F., and Nelson, P. C. 1995. Improving search efficiency using possible subgoals. *Mathematical and Computer Modelling* 22(4-7):397-414.
- Dubois, N., and Semet, F. 1995. Estimation and determination of shortest path length in a road network with obstacles. *European Journal of Operational Research* 83:105-116.
- Hart, P. E., and Nilsson, N. J. 1968. A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions of Systems Science and Cybernetics* 4(2):100-107.
- Korf, R. E. 1985. Depth-first iterative-deepening: An optimal admissible tree search. *Artificial Intelligence* 27:97-109.
- Korf, R. E. 1987. Planning as search: A quantitative approach. *Artificial Intelligence* 33:65-88.
- Korf, R. E. 1996. Artificial intelligence search algorithms. Technical Report TR 96-29, Computer Science Department, UCLA, Los Angeles, USA.
- Kuipers, B. 1978. Modeling spatial knowledge. *Cognitive Science* 2:129-153.
- Pearl, J. 1985. *Heuristics Intelligent Search Strategies for Computer Problem Solving*. Addison-Wesley.
- Pearson, J. 1998. Heuristic search in route finding. Master's thesis, Computer Science Department, University of Auckland, Auckland, New Zealand.
- Rillings, J., and Krage, M. 1992. Travtek: An operational advanced driver information system. In *Proc. Society of Automotive Engineers*, volume P-260, 461-472. Warrendale, PA, USA: SAE.
- Sarkar, U.; Chakrabarti, P.; Ghose, S.; and de Sarkar, S. 1991. Reducing expansions in iterative-deepening search by controlling cutoff bounds. *Artificial Intelligence* 50:207-221.
- Taylor, L. A., and Korf, R. E. 1993. Pruning duplicate nodes in depth-first search. In *Proc. AAAI-93*, 756-761.
- Timpf, S.; Volta, G. S.; Pollack, D. W.; and Egenhofer, M. J. 1992. A conceptual model of wayfinding using multiple levels of abstraction. In *Theories and Methods of Spatio-Temporal Reasoning in Geographic Space*, volume 639, 348-367. Springer Verlag.
- Yang, T.; Shekhar, S.; Hamidzadeh, B.; and Hancock, P. 1991. Path planning and evaluation in ivhs databases. In *Proc. VNIS*, 283-290.