Analysis of Heuristic Search Models

by

Thomas Sudkamp

and

Robert Shanahan

Department of Computer Science
Wright State University
Dayton, OH 43435

Keywords: Automated reasoning, heuristic search

Abstract

A model containing multiple optimal solutions and nonoptimal solutions is constructed to study the performance of A^* heuristic search algorithm. To obtain estimates of the average performance of the algorithm, a domain independent A^* search space is constructed treating the heuristic function, branching factor and number of successful children of a node as random variables. These results are compared to the worst case performance of the models developed by Pohl and Gaschnig. The parameters of the model are defined to simulate the 15 puzzle search space. The results of the 15 puzzle searches are compared with the simulation to determine the effects of the assumptions on the structure of the graph and the heuristic functions.

1. Introduction

Heuristics have been used to inhibit the combinatorial explosion inherent in graph searching. The A^* algorithm [3] is a heuristically guided best first search strategy. Models of search spaces have been constructed to predict the performance of the A^* search. These models utilize simplifying assumptions concerning both the structure of the graph and the heuristic function to facilitate the analysis of the search.

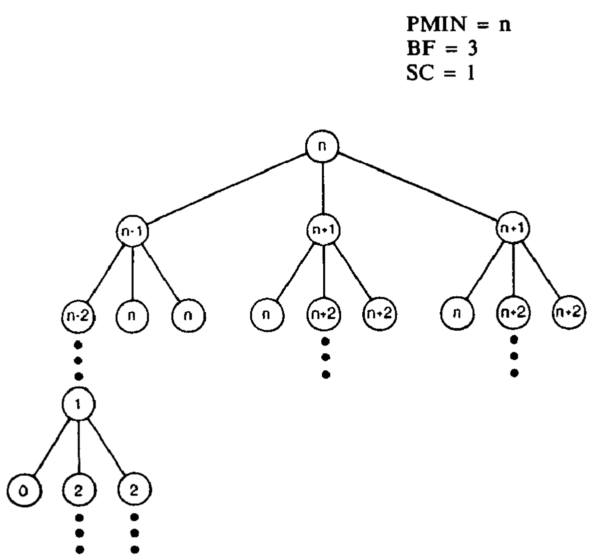
Pohl [6] [7] and Gaschnig [2] developed a model whose search space consists of a tree with uniform branching factor and a single goal node. We will refer to this search domain as the single solution model. A worst case analysis is given for heuristics in which the error is bounded by a fixed constant value and when the error grows linearly with the length of the minimal solution. The consequences of extending this model to include multiple minimal solutions and nonoptimal solutions will be examined.

Pearl [4] used the single solution search graph to analyze the effects of the heuristic function on the search. In this analysis the heuristic function was defined by a distribution over a range of possible values. Following this approach, we construct a domain independent simulation of the A^* search that treats the heuristic, branching factor and number of solutions as random variables over predefined distributions. To determine the influence of the simplifying assumptions on the search, the model is configured to simulate the 15 puzzle search space. Heuristics and search parameters are constructed to compare the performance of the model to results obtained from searches in the 15 puzzle space.

This research was supported by the Air Force Office of Scientific Research/AFSC under contract number F49620-85-C0013 SB5851-0360.

2. Search simulation

A simulation of the A^* algorithm is constructed to study the effects of heuristics and the structure of the graph on the search. The implicit search space is a tree defined by three parameters; the length of a minimal solution path (PMIN), the branching factor (BF) and the number of successful children per node (SC). The arcs of the tree are assumed to have unit length. A node in the tree is characterized by two integer values; its level in the seach tree and the minimal distance to a goal node. Using the A^* terminology, these values are denoted g(N) and $h^*(N)$ respectively. A successful child is a node whose h^* value is one less than that of its parent. The diagram shows the h^* values of the nodes in a search space defined by a uniform branching factor of three, one successful child per node and minimal path length n.



The number of solution paths in the search space can be calculated for constant values of BF and SC using a recurrence relation. We let S(k,n) denote the number of nodes on level k with $h^* = n$ and pos(n) be 1 if n is positive, 0 otherwise.

$$S(0,n) = \begin{cases} 1 & \text{if } n = \text{PMIN} \\ 0 & \text{otherwise} \end{cases}$$

$$S(k+1,n) = SC \cdot S(k,n+1) + pos(n) \cdot (BF-SC) \cdot S(k,n-1)$$

S(k,j) is zero for all values of j outside the interval [minimum(0, PMIN-2 k), PMIN+2 k]. The number of solution paths of length k is given by S(k,0). In a search space with one successful child there are (BF-1) PMIN solution paths of length PMIN+2. If SC = 2, 2^{PMIN} of the BF^{PMIN} nodes on level PMIN are the terminal nodes of minimal solution paths.

A heuristic function h(N) is used to estimate the value $h^*(N)$. This function is defined by a distribution for each h^* value. We assume that the heuristic values of two distinct nodes are conditionally independent. The heuristic function is also assumed to return 0 for all goal nodes.

The search is guided by an evaluation function f that is a linear combination of the g and h values of the node. The scaling factor is a constant w in the interval [0, 1] known as the weight. The evaluation function f will be denoted

$$f(g(N), h(N), w) = (1-w)g(N) + w \cdot h(N)$$

indicating the dependence on h, g and w. Searches using weights 1 and .5 will be refered to as depth first and even weighting respectively.

A node N will be represented by a triple $[h^*(N), g(N), f(g(N), h(N), w)]$. A variation of the A^* algorithm for this search space is given below. We will refer to this search simulation as HSM (Heuristic Search Model).

1. Place [n,0,f(n,0,w)] on OPEN.
2. Choose a node N from OPEN with minimal f value. Remove N from OPEN. If $h^*(N) = 0$, then exit with a solution of length g(N).
3. Determine the branching factor for N, call it j. Determine the number of successful children of N, call it k.

Build k nodes as follows (successful children):

$[h^*(N)-1,\ g(N)+1,\ f(h^*(N)-1, g(N)+1, w)]$

Build j-k nodes as follows (unsuccessful children):

$[h^*(N)+1,\ g(N)+1,\ f(h^*(N)+1, g(N)+1, w)]$

4. Place the j nodes on OPEN.
5. Go to 2.

Steps 3 and 4 constitute the expansion the node N. The list OPEN contains all nodes available for expansion. The standard A^* algorithm maintains an additional list which contains the expanded nodes. The tree structure of the search space guarantees that no node will be rediscovered, removing the need for this additional list.

The number of nodes expanded and the length of the solution paths are used to measure the performance of the search algorithm. The results for a set of search parameters (heuristic, BF and SC distributions and weighting) represent the average over 35 applications of the algorithm. Due to memory restrictions, a search is terminated unsuccessfully when the size of the OPEN list exceeds 100,000 nodes. If one instance of the search terminates unsuccessfully, no results are given for that PMIN value.

3. Bounded error

A heuristic function is said to have bounded error if $h^*(N) - c \leq h(N) \leq h^*(N) + c$ for a constant c. Pohl [6] [7] gives a worst case analysis of the performance of the A^* algorithm guided by this type of heurisitic in the single solution model. The heuristic is assumed to assign a pessimistic value to all nodes on the minimal solution path and an optimistic value to all others; that is, $h(N) = h^*(N) + c$ for all nodes on the minimal solution path and $h^*(N) - c$ for all others. The model also assumes that ties are broken in favor of nongoal nodes. Under these conditions the number of nodes expanded is shown to grow linearly with the length of the minimal solution path for both the even weighting and depth first searches.

To compare the average and worst case performances of the A^* algorithm the search program was exercised with parameters simulating single solution model with constant branching factor 2, bounded error 5 and distribution of successful children defined by

$$SC(N) = \begin{cases} 1 & \text{if } h^*(N) = PMIN - g(N) \\ 0 & \text{otherwise.} \end{cases}$$

Like the worst case analysis [6], the results of the searches with the uniform error distribution exhibit a linear growth of node expansions with path length. The growth rate, however, is much more moderate than that of a completely misleading heuristic. Under these conditions there appears to be little difference in the work required by the even weighting and the depth first strategies.

	even weighting		depth first	
PMIN	Worst case	HSM	Worst case	HSM
25	801	70.9	12801	78.7
50	1601	159.6	25601	159.5
100	3201	343.7	51201	337.6
200	6401	700.0	102401	668.3

4. Bounded error and multiple solutions

Worst case performance estimates are established for the even weighting search when the search space contains multiple optimal and nonoptimal solutions. The same conditions are assumed as in the single solution analysis. The presence additional solutions is shown to force the search to explore additional paths.

Theorem. The worst case performance of A^* with weight bounded error c and constant branching factor has the following properties:

i) If SC is a constant greater than 1, the number of nodes expanded grows exponentially with the minimal solution length.

ii) If SC = 1 then the number of nodes expanded grows polynomially with the minimal solution length. For bounded error c, the number of node expansions is at least $O(PMIN^{2 \cdot c})$.

Proof. A search space with SC >2 has a binary subtree of minimal solution paths. All nodes in this subtree on levels less than PMIN - 2c must be expanded by an even weighting search. Thus the worst case complexity is $O(2^{PMIN})$.

The existence of nonoptimal paths affects the complexity of the search model with only one minimal solution. For c = 1, all paths that contain only one unsuccessful move must be explored to level PMIN. Each node on such a path has an f value less than or equal to that of the minimal goal node and all goal nodes on level PMIN+2. The expansion of a node in the minimal solution path generates BF-1 such paths. A path with one unsuccessful move that diverges from the minimal solution path at level k consists of PMIN-k nodes which must be expanded. The number of such nodes along nonoptimal paths can be computed by

$$\sum_{i=1}^{PMIN} (BF-1)\cdot(PMIN - i)$$

yielding the $PMIN^2$ growth.

For a constant error of 2, paths with 2 unsuccessful moves must be explored. Let N be a node on a path containing one unsuccessful move. There are (BF-1)*k such nodes on level k of the search tree. Paths from N containing one additional unsuccessful move must explored. All nodes with g value < PMIN on such paths will be expanded. Using the previous result, if N occurs on level k there are $(BF-1)\cdot((PMIN-k)^2 - (PMIN-k))/2$ nodes on such paths. Summing over all paths yields $O(PMIN^4)$ growth.

Since paths, not subtrees, are explored the exponential growth is avoided. Following the technique above, estimates for bounded error c can be obtained by considering paths with c or fewer unsuccessful moves.

To obtain a comparison between the average and worst case performance, the bounded error search was simulated on HSM. The heuristic distributions were defined to be uniform within the ranges

$$[\, h^* - c, h^* + c \,] \text{ if } h^*(N) \geq c$$

$$[0, 2\cdot H^*(N)] \qquad \text{otherwise,}$$

where c is the maximal error. The latter condition forces the heuristic function to be nonnegative and return 0 for a goal node.

Figure 1 a) shows the results of a series of searches with bounded error heuristic. These seaches used one, two and a variable number of successful children per node. The number of successful children in the variable search was defined as a uniform distribution between 1 and BF. The results indicate that the exponential growth occurs not only in a worst case scenario, but also when the heuristic error is uniformly distributed over its range.

The depth first results (Figure 1 b)) exhibit a linear growth of node expansions with respect to path length, regardless of the number of solution paths in the graph. The nodes expanded and solution lengths in the searches with a single successful child grow significantly with increases in the heuristic error and branching factor. Increasing the number of solution paths dampens the effects of the other parameters.

An analytic approach for worst case estimates of the depth first strategy appears more complicated than for the even weighting. It is not clear what properties characterize the worst performing heuristic. The heuristic used in the even weighting analysis will lead to a solution with length PMIN+2 in exactly PMIN+2 node expansions.

5. Linearly bounded heuristic

A heuristic is said to be linearly bounded if

$$h^*(N) - b \cdot h^*(N) \le h(N) \le h^*(N) + b \cdot h^*(N),$$

for some constant b between 0 and 1. The single solution model with an even weighting has been shown to have exponential complexity when using a linearly bounded

a) even weighting

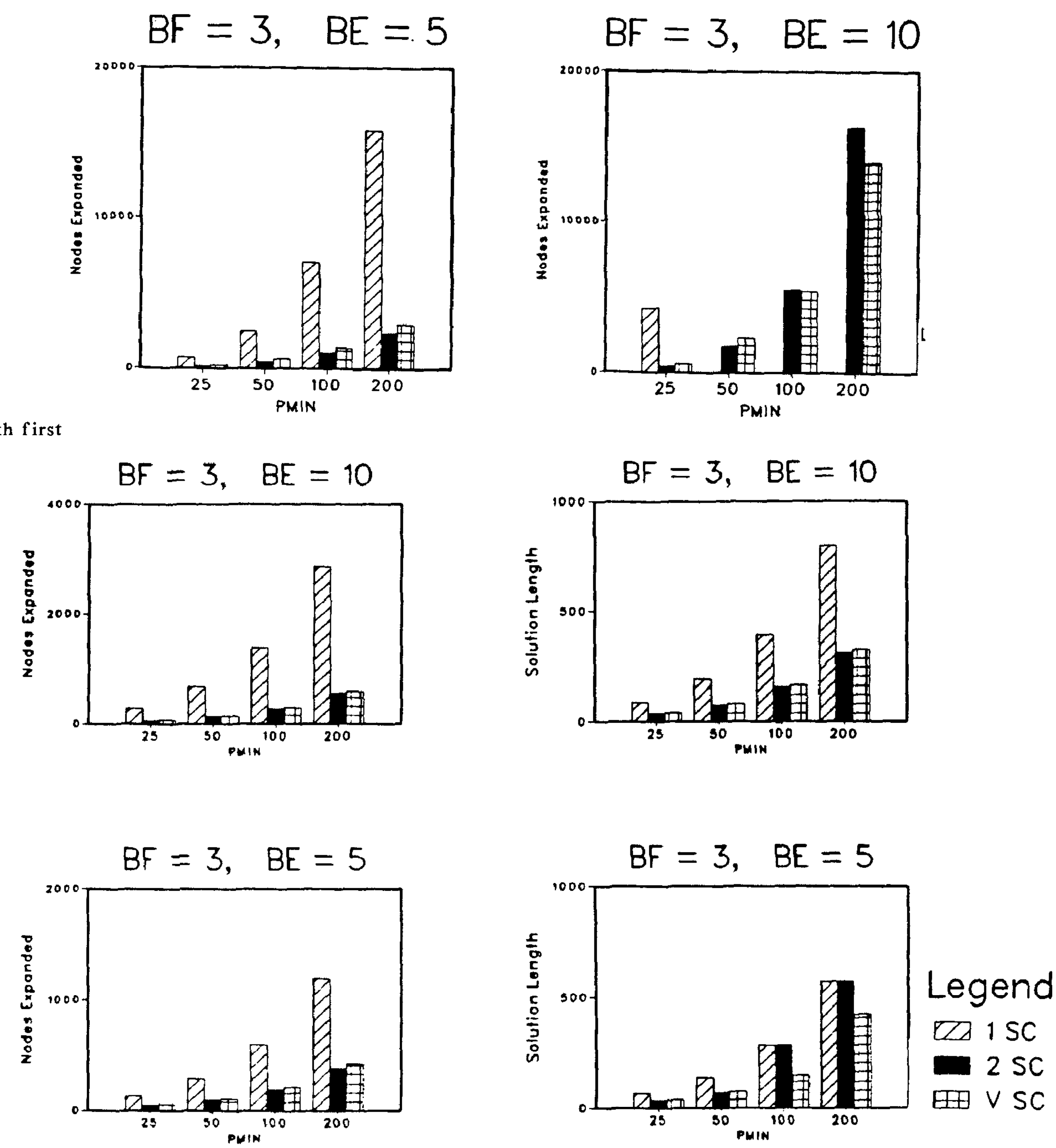


Figure 1. Results of bounded error searches.

heuristic [2]. The extended model can easily be shown to exhibit a similar worst case behaviour.

The model was exercised with one and two successful children per node. In the former case there is only one node on each level lying on a minimal solution path. With two successful children there are 2^n such nodes on level n. To bridge the gap between one and an exponentially growing number of minimal solutions, a search space was defined in which the number of nodes on level n lying on optimal solution paths is n^2.

The results of searches with the heuristic values uniformly distributed within the linear bounds and constant branching factor 3 are given in Table I. The results for the search space with $PMIN^2$ optimal solutions is labelled SQ.

The even weighting search fails to find a solution for path length 200 regardless of the number of solution paths. The additional solution paths causes the generation of the search tree to behave in a breadth first manner.

With the heuristic providing less accurate estimates as the solution length increases, the depth first strategy in a space in which the number of solution paths increase polynomially with the minimal solution length does not avoid the combinatorial explosion. The addition of multiple optimal solutions appears to retain the linear complexity of the search, at least within this range of PMIN values.

6. Effects of domain assumptions.

The models of Pohl, Gaschnig, Pearl and HSM all impose restrictions on the search graph and the heuristic function to facilitate the analysis of the search performance. One such simplification is assuming the search space to be a tree rather than an arbitrary graph. When the heuristic function is defined by a distribution, the conditional independence of the heuristic values is assumed. We will now compare the performance of the model to searches in a "real" domain in an attempt to exhibit the consequences of these assumptions.

For this purpose we consider graphs generated by the 15 puzzle. The 15 puzzle consists of tiles numbered 1 to 15 in a four by four square. The configuration of the puzzle may be altered by sliding an adjacent tile into the unoccupied square. The objective of the puzzle is to manipulate the tiles to obtain a predefined goal configuration.

1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	■

15 puzzle
goal configuration

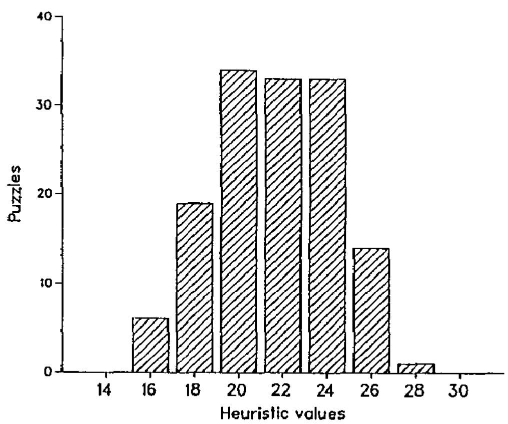
Search space parameters and heuristic distributions are constructed to simulate the 15 puzzle search space. The objective is to compare the model to the results obtained by the A^* algorithm in a domain which includes cycles and nonoptimal solutions.

Two heuristics from the literature [Doran and Michie 1, Pohl 5] will be used to study the 15 puzzle and simulated for the evaluation program.

$h_1(N)$: The sum of the distance of each tile from its currentposition to its position in the goal state, assuming thatthere are no obstacles in this path (Manhattan distance).

$h_3(N)$: The sum over the tiles of the square root of the Manhattan distance from a tile to the blank times the square of the Manhattan distance of the tile from its goal position.

A sample of 15 puzzles was constructed for PMIN values 1 to 18 and even values from 20 to 40. These samples were used to build the distributions necessary to simulate the 15 puzzle heuristics. The distribution of values of the Manhattan distance heuristic for a sample of 140 puzzles with PMIN 30 is given below.



A node with an odd h^* value greater than 17 will use the distribution of previous value increased by one. The distributions generated by the Manhattan distance heuristic underestimate h^*, retaining admissibility when the even weighting is used.

A 15 puzzle search is terminated unsuccessfully if 20,000 nodes are expanded without discovering the goal. The sample with minimal solution length 25 contains a puzzle which is not solved by the depth first, Manhattan distance combination. It must be noted that many puzzles with PMIN > 25 can be solved with fewer than 20,000 node expansions.

The exponential growth of node expansions of the even weighting is evident for both heuristics (Table II). Like the linearly bounded searches, it is not clear whether a weak heuristic (h_1) with a depth first strategy can avoid the exponential explosion. This differs from the conclusions of studies of the 8 puzzle search performance. In this simpler domain Gaschnig [2] observed that the depth first strategy always provided economical solutions.

The search model was exercised with a single, successful child and a branching factor distribution of 25% of the nodes having one child, 50% with two children and 25% with three. A comparison of these results with the 15 puzzle dramatically illustrates the effects of the assumptions of the search performance.

The even weighting searches in the 15 puzzle domain are not greatly effected by the cycles in the search graph. The number of nodes rediscovered and reopened by the search (ReO in Table II) make up less than 10% of the nodes expanded by the search. The simulation exhibits the exponential growth shown by the 15 puzzle with the combinatorial explosion delayed in the model.

For the 15 puzzle domain, depth first searches with large PMIN values reopen more than half of the nodes expanded. For the larger PMIN values, the nodes on the solution path in the 15 puzzle space make up less than 5% of those expanded while in the simulation over half of the expanded nodes lie on the solution. The acyclic model does not permit this type of behaviour, greatly simplifying the depth first approach. In this case it appears that depth first models, both worst case or average using a tree for the underlying graph, do not accurately reflect the complexity of searches in a more complicated domain.

7. Future work

The comparison of the 15 puzzle results to the results of the model vividly illustrates the differences between simplified models and general graph searching. One direction of research is to refine the model to more accurately reflect various domains. An assumption whose effect can be evaluated with only a minimal change to the existing model is the conditional independence of the heuristic function. The heuristic distribution must be defined for the h^* value of the node and the h value of its parent. The Mahanattan distance heuristic exhibits this dependence of heuristic values in the 15 puzzle domain. The h value of a child can differ from that of its parent by at most 1.

PMIN	15 puzzle Exp	15 puzzle Path	15 puzzle ReO	HSM Exp	HSM Path
10	12.5	10.0	0.0	11.1	10.0
15	39.4	15.0	0.0	20.7	15.0
20	188.9	20.0	0.0	40.1	20.0
25	1437.5	25.0	0.0	138.7	25.0
30	3869.5	30.0	0.0	696.1	30.0
35	-	-	-	3812.5	35.0
40	-	-	-	22934.0	40.0

h_1 : even weighting (admissible).

PMIN	15 puzzle Exp	15 puzzle Path	15 puzzle ReO	HSM Exp	HSM Path
10	62.9	15.8	16.9	10.9	10.2
15	461.0	43.5	196.3	19.5	17.0
20	1418.7	97.0	561.2	32.4	26.1
25	-	-	-	48.3	35.3
30	-	-	-	66.3	44.7
35	-	-	-	90.0	56.5
40	-	-	-	113.1	66.9

h_1 : depth first.

PMIN	15 puzzle Exp	15 puzzle Path	15 puzzle ReO	HSM Exp	HSM Path
10	12.5	10.0	0.1	11.3	10.9
15	54.9	16.5	1.6	20.5	16.7
20	282.2	25.2	8.2	33.6	23.4
25	894.7	33.1	55.1	51.7	31.5
30	1880.1	42.7	136.8	73.9	40.8
35	-	-	-	91.1	47.1
40	-	-	-	110.1	56.0

h_3 : even weighting.

PMIN	15 puzzle Exp	15 puzzle Path	15 puzzle ReO	HSM Exp	HSM Path
10	28.8	14.0	0.6	11.3	10.9
15	650.1	37.6	350.9	19.0	17.1
20	1230.4	69.3	578.9	32.2	25.5
25	2326.6	119.5	1146.4	46.8	34.0
30	3807.7	123.0	2177.3	65.1	43.7
35	3928.7	154.5	2110.0	80.3	53.1
40	4119.4	176.0	2298.4	101.4	62.0

h_3 : depth first.

Table II. 15 puzzle and HSM results.

References

1. Doran, J., and Michie, D., "Experiments with the graph traverser program", Proceedings of the Royal Society, vol. 294(A), pp. 235 - 259, 1966.

2. Gaschnig, J., Performance Measurement and Analysis of Certain Search Algorithms, Ph.D. Dissertation, Computer Science Department, Carnegie-Mellon University, 1979.

3. Hart, P. E., Nilsson, N. J. and Raphael, B., "A formal basis for the heuristic determination of minimum cost paths", IEEE Transactions on Systems, Science and Cybernetics, vol. 4, number 2, 1968, pp. 100 - 107.

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

5. Pohl, I., Bi-directional and Heuristic Search in Path Problems, Report Number 104, Stanford Linear Accelerator Center, Stanford, California, 1969.

6. Pohl, I., "First results on the effect of error in heuristic search", in Machine Intelligence 5 (ed. B. Meltzer and D. Michie), American Elsevier, pp. 219 - 236, 1970.

7. Pohl, I., "Practical and theoretical considerations in heuristic search algorithms", in Machine Intelligence 8 (ed. E. Elcock and D. Michie), Wiley, pp. 55 - 72, 1977.