KYBERNETIKA — VOLUME 7 (1971), NUMBER 6

# On an Evaluation Function for Heuristic Search as a Path Problem*

ĽUDOVÍT MOLNÁR

The new evaluation function for heuristic search as a path problem is proposed and compared with those proposed earlier. The comparison is done for special problem space and the worst case. The theoretical and experimental results give us the justification for the proposed evaluation function.

## INTRODUCTION

In a process of a finding a path through a graph consisting of a set of nodes corresponding to discrete states of a solved problem we try to utilize all knowledge from a given problem. The Graph Traverser [1] uses a purely heuristic evaluation function $f(n) = h(n)$ derived from a property of a given problem to control search for the goal node as an estimate of a distance from a node $n$ to the goal node.

Hart, Nilsson and Raphael [4] have proposed a compound evaluation function $f(n) = g(n) + h(n)$ where $g(n)$ is a current distance from the start node to a node $n$ found during search and $h(n)$ is a heuristic component which estimates the distance from a node $n$ to the goal node.

Pohl [7] has done experiments with a weighting of parameters for each part of the compound evaluation function and discovered that improvement in a solution can be reached for some values of parameters.

Similar experiments have been done also by Michie and Ross [5] who have done an optimalization of parameters of the heuristic component of an evaluation function.

Molnár [6] has done an analysis of parameters of an evaluation function and discovered that parameters should be not constants but functions which change their values during search.

* The paper was partially written while the author was the Visiting Research Worker at the Department of Machine Intelligence and Perception University of Edinburgh as the student of the British Council.

In all these works a better utilization of a property of a given problem was tried to do. Next we will show another kind of an utilization of a property of a given problem by adding to an evaluation function so called difference of heuristic parts of an evaluation function. We will also prove in the worst case analysis that this evaluation function is better in this sense than an evaluation function without difference.

## THE PROBLEM SPACE AND THE ALGORITHM

We will use the same problem space and the algorithm as in [6], except an evaluation function which will be defined latter.

### The evaluation function

The evaluation functions used in heuristic search can be classified as follows:

$$f(n) = g(n) \tag{1}$$

– exhaustive parallel or bredth first search – no heuristic information is used,

$$f(n) = h(n) \tag{2}$$

– pure heuristic search [1],

$$f(n) = g(n) + h(n) \tag{3}$$

– compound heuristic search [4],

$$f(n) = \omega\, g(n) + \omega'\, h(n) \tag{4}$$

– compound heuristic search with constant parameters [7],

$$f(n) = \omega(n)\, g(n) + \omega'(n)\, h(n) \tag{5}$$

– compound heuristic search with function parameters [6].

We add to this list the new type of an evaluation function:

$$f(n) = \omega(n)\, g(n) + \omega'(n)\, h(n) + D(n)\,, \tag{6}$$

where $\omega(n)$, $\omega'(n)$ are function parameters, $g(n)$ is a number of steps from the start node to a node $n$, $h(n)$ is an estimate of a number of steps from a node $n$ to the goal node,

$$D(n) = \omega'(n)\, h(n) - \omega'(n_p)\, h(n_p)\,,$$

where $n_p$ is the parent (predecessor) of $n$. If $n$ is the root then

$$D(n) = 0\,.$$

This evaluation function will be compared with evaluation functions of the form (5). For a simplicity parameters will be used as constants equal 1.

**The comparison of evaluation functions of the forms (5) and (6)**

As we have said the comparison will be done for the worst case and we will use the easy analysable problem space the regular binary tree.

Let $h'(n)$ be a perfect estimator, $h(n)$ be a given heuristic function, $\varepsilon$ be a bound on the error 0, 1, 2, ...,

$$h'(n) - \varepsilon \leqq h(n) \leqq h'(n) + \varepsilon\,.$$

To make $h(n)$ as bad as possible we add $\varepsilon$ to each node on the shortest solution path and subtract $\varepsilon$ from each node off the shortest solution path.

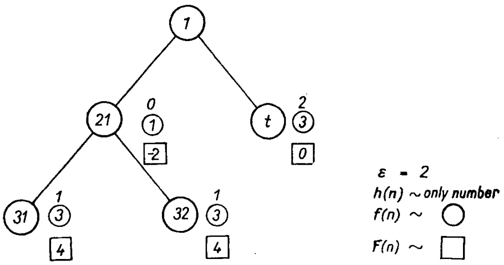

Fig. 1.

At first we will give an intuitive justification of the "difference" concept of an evaluation function. Consider a tree as in Fig. 1 with $\varepsilon = 2$ and monitor our algorithm using as evaluation functions

$$f(n) = g(n) + h(n)\,,$$

$$F(n) = g(n) + h(n) + D(n)\,.$$

Let $t$ be the goal node. Then

$$h'(t) = 0\,; \quad g(t) = 1\,;$$

$$h(t) = h'(t) + \varepsilon = 2\,;$$

$$f(t) = g(t) + h(t) = 3\,;$$

$$h(21) = h'(21) - \varepsilon = 0\,;$$

$$h(31) = 1\,;$$

$$F(t) = g(t) + h(t) + h(t) - h(1) = 0\,;$$

$$f(21) = 1\,; \qquad F(21) = -2\,;$$
$$f(31) = 3\,; \qquad F(31) = \;\; 4\,.$$

When we now compare the values of $f(21)$ and $F(21)$ with the values of their successors i.e. $f(31)$ and $F(31)$ we can see that there is a greater difference between $F(21)$ and $F(31)$ than between $f(21)$ and $f(31)$. This means that if we use as an evaluation function $F(n) = g(n) + h(n) + D(n)$ the search along an incorrect path will be terminated sooner than if we use a function without difference $D(n)$.

Now we will prove it formally.

**Theorem 1.** *Let $k$ be the distance from the start node to the goal node and*

$$f(n) = g(n) + h(n)$$

*be the evaluation function. Then the maximum number of nodes visited in a binary tree will be*

$$2^{\varepsilon+1} \,.\, k + 1\,.$$

Proof. We must visit in our binary tree all nodes on the solution path and all nodes off the solution path whose value is less (in the case of ties i.e. nodes with a same value of an evaluation function, we choose, in the sense of the worst case analysis, "the worst" node) than the value of the goal node.

If $t$ is the goal node then

$$f(t) = g(t) + h(t) = g(t) + h'(t) + \varepsilon = k + \varepsilon\,.$$

For any node $n$ off the solution path is

$$f(n) = g(n) + h(n) = g(n) + h'(n) - \varepsilon\,.$$

If $w$ is a number of steps off the solution path, then (see [6])

$$g(n) + h'(n) = k + 2w$$

and we can write

$$f(n) = 2w + k - \varepsilon\,.$$

As we have said the relation between $f(n)$ and $f(t)$ must be

$$f(n) > f(t)\,,$$
$$2w + k - \varepsilon > k + \varepsilon\,,$$
$$2w > 2\varepsilon\,,$$
$$w > \varepsilon\,.$$

Since we want to visit the minimum number of nodes we let

$$w = \varepsilon + 1\,.$$

Since the number of leaves in the binary tree grows as $2^{w-1}$ the number of nodes visited is equal:

$$k + 1 + k + 2k + 2^2k + \ldots + 2^{w-1}k = 1 + 2k + k\sum_{i=1}^{w-1} 2^i =$$

$$= 1 + 2^w k = 1 + 2^{\varepsilon+1}k\,.$$

**Theorem 2.** *Let $k$ be the distance from the start node to the goal node and*

$$f(n) = g(n) + h(n) + D(n)$$

*be the evaluation function. Then the maximum number of nodes visited in a binary tree will be*

$$2^\varepsilon \,.\, k + 1\,.$$

Proof. In this case we must also visit all nodes on and off the solution path whose value is less than the value of the goal node.

If $t$ is the goal node then

$$f(t) = g(t) + h(t) + D(t) = g(t) + h'(t) + \varepsilon + h'(t) + \varepsilon - h'(t_p) - \varepsilon =$$

$$= k + \varepsilon - 1\,.$$

For any other node $n$ off the solution path we have two possibilities:

1. A node $n$ is 1 step off the solution path, when

$$f(n) = g(n) + h(n) + D(n) = g(n) + h'(n) - \varepsilon + h'(n) - \varepsilon - (h'(n_p) + \varepsilon)\,.$$

Since the node $n_p$ is on the solution path we had to use "$+\varepsilon$" in $D(n)$.

$$f(n) = g(n) + h'(n) - 3\varepsilon + h'(n) - h'(n_p)\,.$$

Using the relation

$$g(n) + h'(n) = k + 2w$$

as in Theorem 1 we will get

$$f(n) = k + 2w - 3\varepsilon + 1\,.$$

This value must be greater than $f(t)$ i.e.:

$$f(n) > f(t)\,,$$

$$k + 2w - 3\varepsilon + 1 > k + \varepsilon - 1\,,$$

$$2w > 4\varepsilon - 2\,,$$

$$w > 2\varepsilon - 1\,.$$

Since $n$ is 1 step off the solution path

$$w = 1$$

and for $\varepsilon$ we can write

$$\varepsilon < 1 .$$

This means that all nodes 1 step off the solution path will have $\varepsilon < 1$ (i.e. $\varepsilon = 0$) for which the value of $f(n)$ is greater than that of $f(t)$. This conclusion is same as that of Theorem 1.

The number of nodes visited is equal

$$1 + k = 1 + 2^{\varepsilon} . k .$$

2. A node $n$ is more than 1 step off the solution path. Then

$$f(n) = g(n) + h(n) + D(n) = g(n) + h'(n) - \varepsilon + h'(n) - \varepsilon - h'(n_p) + \varepsilon =$$
$$= k + 2w - \varepsilon + 1$$

and again

$$f(n) > f(t) ,$$
$$k + 2w - \varepsilon + 1 > k + \varepsilon - 1 ,$$
$$2w > 2\varepsilon - 2 ,$$
$$w > \varepsilon - 1 .$$

Since we want to visit the minimum number of nodes we let

$$w = \varepsilon$$

and for the number of nodes visited we can write:

$$1 + 2^{\varepsilon} . k .$$

The above results are extendable to any tree with a unique goal node.

**Theorem 3.** *If our algorithm is searching a tree for the goal node, then*

1. *if we use as an evaluation function*

$$f(n) = g(n) + h(n)$$

*we will visit at least as many nodes as if we use an evaluation function*

$$f(n) = g(n) + h(n) + D(n)$$

*in the sense of the above worst case analysis;*

2. *if the error bound on $h_1(n)$ is $\varepsilon_1$ and $\varepsilon_2$ is on $h_2(n)$ with the relation $\varepsilon_1 < \varepsilon_2$ then in the sense of the worst case analysis with $h_2(n)$ must be visited more nodes than with $h_1(n)$.*

Proof. The proof follows from Theorems 1 and 2.

## EXPERIMENTS

Our experiments have been carried out with the Fifteen puzzle. As an evaluation function we used

$$f(n) = \omega(n)\, g(n) + \omega'(n)\, h(n) + D(n)$$

where $\omega(n)$, $\omega'(n)$ are function parameters [6] used in the form

$$\omega(n) = 1\,,$$

$$\omega'(n) = \frac{h(n) + \varepsilon(n) + 2}{h(n) - \varepsilon(n)}$$

where

$$\varepsilon(n) = R(n) = \sum_{i=1}^{15} p_i$$

defined as in [6].

This evaluation function was tested on 50 randomly generated Fifteen puzzles. The size of the partial tree was 200 nodes, resignation occured when 500 nodes had been encountered.

The results were compared with those obtained using an evaluation function

$$f(n) = \omega(n)\, g(n) + \omega'(n)\, h(n)\,.$$

In both evaluation functions

$$h(n) = \sum_{i=1}^{15} p_i^2$$

where $p_i$ is a number of steps of the $i$-th piece from the goal position. The experimental results are summarized in Table 1.

**Table 1.**

| | Sample size | % of puzzles solved | The average number of nodes encountered/puzzle |
|---|---|---|---|
| without difference $D(n)$ | 50 | 48 | 402 |
| with difference $D(n)$ | 50 | 60 | 364 |

## CONCLUSION

The modification of an evaluation function for heuristic search as a path problem is discussed. The purpose of the modification is the better utilization of a knowledge of a given problem. The comparison with evaluation functions defined earlier have been done for the special problem space and for the worst case but experiments which have been done showed the applicability of the theoretical results to the real problem space too.

*Acknowledgements.* The author wishes to thank to Professor Donald Michie — Professor of the Department of Machine Intelligence and Perception University of Edinburgh, under whose supervision the work was done, Dr. R. Kowalski and Mr. R. Ross of the same Department for many helpfull discussions and criticisms.

(Received February 22, 1971.)

### REFERENCES

[1] Doran, J. E., Michie, D.: Experiments with the Graph Traverser program. Proc. R. Soc. *294* (1966), 1437, 235—259.

[2] Doran, J. E.: An approach to automatic problem solving. Machine Intelligence 1 (eds. N. L. Collins, D. Michie). Oliver and Boyd, Edinburgh 1967.

[3] Doran, J. E.: New developments of the Graph Traverser. Machine Intelligence 2 (eds. E. Dale, D. Michie). Oliver and Boyd, Edinburgh 1968.

[4] Hart, P., Nilsson, N., Raphael, B.: A formal basis for the heuristic determination of minimum cost paths. I.E.E.E. Trans. on Sys. Sci. and Cybernetics *4* (1968), 2, 100—107.

[5] Michie, D., Ross, R.: Experiments with the Adaptive Graph Traverser. Machine Intelligence 5 (eds. B. Meltzer, D. Michie). Edinburgh University Press, Edinburgh 1970.

[6] Molnár, L.: On parameters of an evaluation function for heuristic search as a path problem. Kybernetika *7* (1971), 5, 386—393.

[7] Pohl, I.: First results on the effect of error in heuristic search. Machine Intelligence 5 (eds. B. Meltzer, D. Michie). Edinburgh University Press, Edinburgh 1970.

# O vyhodnocovacej funkcii pre heuristické hľadanie ako problém cesty

ĽUDOVÍT MOLNÁR

V článku sa pojednáva o vyhodnocovacej funkcii pre heuristické hľadanie ako problém hľadania cesty. Navrhuje sa nový typ vyhodnocovacej funkcie, ktorá efektívnejšie využíva znalosť problému. Je porovnávaná pre najhorší prípad s vyhodnocovacími funkciami navrhnutými skôr. Teoretické i experimentálne výsledky ukazujú, že táto vyhodnocovacia funkcia je lepšia, než vyhodnocovacie funkcie navrhnuté skôr.

*Ľudovít Molnár, prom. fyz., Katedra matematických strojů EF SVŠT (Department of Computer Science — Slovak Technical University), Vazovova 1/b, Bratislava.*