The implication of the A* and PEA* algorithms in the 8-puzzle problem.*

Arroni del Riego, Sergio[276341], Galán Freire, Alejandro[277346], and Antón de la Calle, Manuel[276213]

Oviedo University, Oviedo Asturias, Spain
https://uniovi.es

Abstract. Search algorithms are very important in Artificial Intelligence because they are the basis for many of the solutions in the different fields of Artificial Intelligence. In this work we will explain the A* and PEA* algorithms and we will compare them in the 8-puzzle problem. We will explain the differences between them. In addition, we will carry out an experimental study on the different heuristics we have used, which use the A* and PEA* algorithms. Finally, we will give our conclusion on the data obtained and we will say which is for us the better of the two algorithms.

Keywords: A* · PEA* · Search · heuristic · 8-puzzle · IA · Uniovi.

1 Introduction

In this section, we will introduce the subject to be dealt with as well as a brief description of the rest of the sections of the work.

1.1 Description of the topic to be addressed

In this work we are going to apply the algorithm A* and PEA* to the 8-puzzle problem, for this we are going to use different heuristics, one of these heuristics is the one we propose in this work, which we call "The most humble first" or MHF.

1.2 Description of the sections of the work

In the following points we will discuss:

2. Description of the 8-puzzle problem, in this section we will detail how the problem in question works.
3. Description of the algorithms involved in the work, in this section we will explain in detail how the algorithms used work, as well as a brief state of the art of them.

* Degree in Software Engineering, University of Oviedo

4. Application of the algorithms to the 8-puzzle problem, we will explain the approach we have taken to the 8-puzzle problem, as well as the heuristics to be used, including the one we propose in this work.
5. Experimentation, we will compare different heuristics, as well as the one proposed in this wok.
6. Conclusion, here we will give our critical opinion on this application of the algorithm.

2 The 8-puzzle problem

The 8-puzzle problem is well known and used for the study of search algorithms, informed or uninformed, generally of the second type, i.e., they also include the study of various heuristics that try to reach the objective state in the optimal way. However, heuristics can be applied simply with the intention to analyze it without the need for it to be good.

2.1 Description of problem

It is actually a concrete example of the n-puzzle problem, which is used because it is simple to understand at this size.

So, for this particular example, we have a 3x3 board where 8 of the cells are filled with all the numbers from 1 to 8 while one remains empty. This will be the one to which we have to move the cells in order to reach the desired final state.

There are 4 possible movements, displacement of a cell to its adjacent cell located in the north, south, east or west as long as the cell is empty in one of the locations.

The cost of each movement is 1, and the goal is to reach the target state by adding as few movements as possible and, therefore, achieving the lowest cost.

The target state that we will consider is as follows: 1, 2, 3, 8, 0, 4, 7, 6, 5

2.2 Classic methods of resolution

As I said at the beginning, the algorithms that can be used to solve this problem can be informed, such as algorithms A*, PEA*, IDA*, and not informed such as DFID, BFS....

We will study the application of the A* and PEA* algorithms with heuristics having different characteristics.

3 The search algorithms

In this section, we will discuss about search algorithms, doing a distincion between uninformed and informed search algorithms. The uninformed algorithms that we will discuss are: Best First (BF). The informed algorithms that we will discuss are: A* and PEA*.

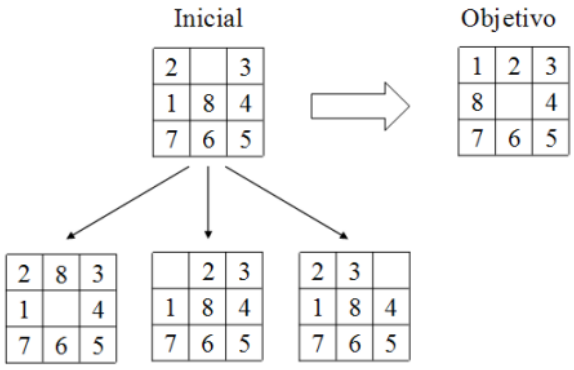

Fig. 1. Figure showing the starting point, the objective and the stages we went through to achieve it.

3.1 Best first search (BF)

The uninformed search algorithms are those thay do not use any information about the problem, they just expand the next node in the frontier. The most important thing that differentiates the various uninformed algorithms is the method they use to expand the next node in the frontier. Hereafter, we will trate the Best First (BF) search algorithm.

We will start with the "Best First" (BF) search algorithm [1], which is the basis of informed search algorithms, such as A^*. BF is based on the idea of idea of expanding the node with the lowest cost. The cost of a node is calculated by the heuristic function $h(n)$ used.
In the BF algorithm, we have two list, the list "open" that this list contains the nodes that are not expanded yet, and the list "closed" that contains the nodes that are expanded:

1. Initialises the "open" list with the start node.
2. Initialises the "closed" list with the empty list.
3. While the "open" list is not empty:
   (a) Select the node with the lowest cost, following the given $h(n)$, in the "open" list and remove it from the "open" list.
   (b) If the node is the goal, stop.
   (c) If not, add the node to the "closed" list and expand its children.
   (d) For each child:
      i. If the child is in the "closed" list, do nothing.
      ii. If the child is not in the "open" list, add it.
      iii. If the child is in the "open" list, but the path to the child is better than the previous path, replace the child in the "open" list with the new child.
4. Return to Step 3.

You can see a more detailed description and a implementation of the algorithm in Russell's and Norvig's book [2].

3.2 The A^* algorithm

The informed search algorithms are those that use the any information about the problem to expand the next node in the frontier.
The A^* algorithm is a variation of the BF algorithm, this was proposed by Peter E. Hart, Nils J. Nilsson and Bertram Raphael in their work in 1968 [3].
In A^*, use a function f to evaluate the nodes, this function in A^* is representate by $f^*(n)$ and this is the cost of the shortest path from initial to n conditional on passing through n. The function $f^*(n)$ (where n is any node) is defined as equation 1:

$$f^*(n) = g^*(n) + h^*(n) \tag{1}$$

Where $g^*(n)$ is the cost of the path from the initial node to the node n, and $h*(n)$ is the heuristic function that estimates the cost of the path from the node n to the goal node, you can see this function in more detail in Nilsson's book "Principles of artificial intelligence" [4]. In A* we talk about f*, g* and h*, but in most cases these are only estimates because it is very complicated for complex problems to know the exact values, if we knew them the algorithm would go straight to the goal. Instead we use the estimates f, g and h, so the function would be as we can see in equation 2:

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

The A^* algorithm is as follows, we have two list, the list "open" that this list contains the nodes that are not expanded yet, and the list "closed" that contains the nodes that are expanded:

1. Initialises the "open" list with the start node.
2. Initialises the "closed" list with the empty list.
3. While the "open" list is not empty:
   (a) Select the node with the lowest cost, you can se the equation in 1, in the "open" list and remove it from the "open" list.
   (b) If the node is the goal, stop.
   (c) If not, add the node to the "closed" list and expand its children.
   (d) For each child:
      i. If the child is in the "closed" list, do nothing.
      ii. If the child is not in the "open" list, add it.
      iii. If the child is in the "open" list, but the $f(child)$ is better than the previous path, replace the child in the "open" list with the new child.
4. Return to Step 3.

One problem that have the A^* algorithm is that it can be very slow, because it can expand a lot of nodes, and this can be a problem if the problem has a lot of nodes. To solve this problem, we can use the PEA* algorithm.
If you want to see a more detailed description and a implementation of the algorithm, you can see Russell's and Norvig's book [2].

3.3 The PEA^* algorithm

The PEA^* algorithm is a variation of the A^* algorithm, in fact it is faster than the base algorithm A^*. PEA^* is a not admitted algorithm, this means that it is not guaranteed to find the optimal solution, but it is very fast and it is very useful in practice. The PEA^* algorithm have a function, very similar to the A^* function, but it is not the same, this function is called $f^*_{PEA}(n)$ and is defined in equation 3:

$$f^*_{PEA}(n) = g^*(n) + h^*(n) * (1 + \epsilon) \tag{3}$$

Where $g^*(n)$ is the cost of the path from the initial node to the node n, $h*(n)$ is the heuristic function that estimates the cost of the path from the node n to the goal node and ϵ is a constant that is used to control the expansion of the nodes.
You can see this algorithm in more detail in Maria Rita's tesis [5].

4 Application of the A* and PEA^* algorithms to the 8-puzzle problem

The two algorithms that we are going to study are informed, i.e., we are facing a heuristic search and, therefore, we have to apply a heuristic to reach the solution. The state space of this problem is defined by the set of possible combinations on the board, i.e. the location of all the tiles.

A* is admissible when an admissible heuristic is applied to it, but PEA* is an $\varepsilon - admissible$ algorithm, which are algorithms that give up admissibility when the problem size is too large. Therefore, for the study of this algorithm applied to the 8-puzzle problem we will use different values of epsilon to be specified in the experimental study.

We will apply the same heuristics to both algorithms. They are the following:

1. h1(n): number of tiles that are out of place. Easy to perform, but does not provide the best possible results. Monotone.
2. h2(n): sum of orthogonal distances from each tile to its final position. Effective for short problems giving the best solutions. Monotone.
3. h3(n): 2*number of tiles at orthogonal distance 2 from their final position
4. h4(n): h2 multiplied by 0.4 summed with h1

4.1 Properties of heuristics

The first two heuristics are typical for solving the 8-puzzle problem with the A* algorithm. They are monotone by satisfying that their sequence of values f(n) of the expanded nodes is non-decreasing, and they also satisfy that

$$h(n1) \leq h(n2) + c(n1, n2)$$

, and this also implies that they are admissible, i.e., they always reach the optimal solution.

On the other hand, we know that h2 is the best heuristic since the ideal case is that the length of the trajectory to the target position is equal to the orthogonal distance of the token to its target. Also this heuristic is more informed than the rest, i.e., h2 dominates the rest of heuristics tested for solving this problem, which means that all nodes expanded by h2 will be expanded by the rest of heuristics. It satisfies

$$hx(n) < h2(n) \leq h^*(n)$$

On the other hand, we believe that h3 also meets the characteristics of the first two heuristics because of the data obtained in the study shown in the next point of the practice. Anyway, it is not as good heuristic as h1 and h2.

Finally, h4 is not monotonic because the f(n) values of the expanded nodes is decreasing. Moreover, it is not admissible and we can make the following proof:

We know that h1 and h2 are admissible so they satisfy that

$$hx(n) \leq h^*(n)$$

for all values of n. If we then assume that h1(n) = h2(n) = h*(n) = 1, we have that

$$h4(n) = 1 + 0.4 * 1 > h * (n)$$

, so it does not satisfy the admissibility condition.

5 Experimental study

TODO

6 Conclusions

As a conclusion we can observe in the tests performed that the best algorithm is A* as it always finds the optimal solution, while sometimes PEA* does not achieve it (not even reaching the target state) increasing the number of moves to perform and the cost while decreasing the time spent as well as the number of expanded nodes. We also note that among the epsilon values tested (0.1, 10, 100) the one that gives the best results is 10.

As for the heuristics, according to the study carried out, the best is h2, expanding fewer nodes than the rest as it is more informed and always obtaining the optimal solution, the latter as well as h1 and h3. The worst of all we see is h4, not being admissible and in the results not reaching the best possible solution in one of the occasions.

7 DE AQUI PARA ABAJO = BASURA, LO DEJO DE EJEMPLO

Please note that the first paragraph of a section or subsection is not indented. The first paragraph that follows a table, figure, equation etc. does not need an indent, either.

Subsequent paragraphs, however, are indented.

Sample Heading (Third Level) Only two levels of headings should be numbered. Lower level headings remain unnumbered; they are formatted as run-in headings.

Sample Heading (Fourth Level) The contribution should contain no more than four levels of headings. Table 1 gives a summary of all heading levels.

Table 1. Table captions should be placed above the tables.

Heading level	Example	Font size and style
Title (centered)	**Lecture Notes**	14 point, bold
1st-level heading	**1 Introduction**	12 point, bold
2nd-level heading	**2.1 Printing Area**	10 point, bold
3rd-level heading	**Run-in Heading in Bold.** Text follows	10 point, bold
4th-level heading	*Lowest Level Heading.* Text follows	10 point, italic

Displayed equations are centered and set on a separate line.

$$x + y = z \tag{4}$$

Please try to avoid rasterized images for line-art diagrams and schemas. Whenever possible, use vector graphics instead (see Fig. 1).

Theorem 1. *This is a sample theorem. The run-in heading is set in bold, while the following text appears in italics. Definitions, lemmas, propositions, and corollaries are styled the same way.*

Proof. Proofs, examples, and remarks have the initial word in italics, while the following text appears in normal font.

For citations of references, we prefer the use of square brackets and consecutive numbers. Citations using labels or the author/year convention are also acceptable. The following bibliography provides a sample reference list with entries for journal

References

1. J. Pearl, "Heuristics : intelligent search strategies for computer problem solving," *Addison-Wesley Series in Artificial Intelligence*, p. 382, 1984.
2. P. R. Norvig and S. A. Intelligence, *A modern approach*, vol. 90. Prentice Hall Upper Saddle River, NJ, USA: Pearson Education, Inc., 2002.
3. P. E. Hart, N. J. Nilsson, and B. Raphael, *A formal basis for the heuristic determination of minimum cost paths*, vol. 4. IEEE, 1968.
4. N. J. Nilsson, *Principles of artificial intelligence*. Springer Science & Business Media, 1982.
5. M. R. Sierra Sanchez *et al.*, "Mejora de algoritmos de busqueda heuristica mediante poda por dominancia. aplicacion a problemas de scheduling," *Universidad de Oveido*, 2009.

Annex 1: Work Distribution

Arroni del Riego, Sergio[$Uo276341$]: I realize the Introduction, A*, PEA*, and BF algorithms explication, Abstract, and KeyWords.
Galán Freire, Alejandro[$Uo277346$]: I realize the 8-puzzle problem explication, the application of the A* and PEA* algorithms to the 8-puzzle problem, and conclusions.
Antón de la Calle, Manuel[$Uo276213$]: I realize the Experimental study.
All: All of us did the tasks of writing in English and using LaTex for the writing of the work.