Al-Mukhtar Journal of Sciences 34 (2): 91-104, 2019

The Choice of Exhaustive or Heuristic Search to Solve AI Problems

Dalya A. Gatsh
Computer Science Department, Faculty of Arts and Sciences, Omar Al-Mukhtar University, Derna-Libya

Received: 02 April 2019/ Accepted: 05 September 2019

Doi:

Abstract: Artificial intelligence is an appealing area of research in computer science because it is concerned with the discovering of effective techniques that have been mainly motivated from human beings or their living environments to solve problems that have special nature. In this research, we aim first to introduce and analyze the common characteristics of problems that artificial intelligence interested in, and then we will highlight how to prepare such problems to solve them by search. The main goal of our study is helping us to decide which search strategy is better through investigating the behavior of most popular search strategies to find out the desired solution for two examples of a simple artificial intelligence problem. Our experiments presented that the required time and memory space to solve the problem mainly affected by many factors such as the applied search mechanism, the solution position, the number of available solutions, and the complexity in search.

Keywords: AI problem, Search problem, Exhaustive search, Heuristic search.

INTRODUCTION

While computer science ,as a research area, is concentrated around the automation of routine and complex humans tasks to make their daily life comfortable and rapid, the branch of Artificial Intelligence (AI) is concerned with finding out applicable ways to solve problems that have a special nature and informally called AI problems. In contrast to traditional problems, AI problems lack an obvious algorithm to solve them (Konar, 2000). In other words, artificial intelligence is interested in problems that intelligent humans cannot tell us exactly *how are they going to solve them*, nor *how did they extract the applied strategy?* Therefore, we cannot classify some problems that require intelligence to solve them as AI problems, because they have a straightforward solving algorithm such as the popular River-Crossing puzzles. The absence of a satisfactory algorithm for AI problems is an outcome of many reasons, which are *first*; solving an AI problem includes a huge number of possibilities that we should take into account within the suggested algorithm to be useful. In 1950, Claude Shannon stated that a typical chess game involved about 10^{120} possible moves would require 3×10^{106} years to make the first move using a computer that can examine one move per microsecond (Negnevitsky, 2005). However, such an algorithm conflicts with human nature that always looks for concise, simple, and fast ways to solve a problem. *Second*, artificial intelligence problems may include communicating with an opponent or a dynamic environment such as two-player games or the navigational planning problem (Konar, 2000).

Because of the unexpected reactions, we cannot specify precisely the required steps to solve an AI problem in prior. *Finally*, the scope of most AI problems is not completely known in

***Corresponding Author:** Dalya A. Gatsh *dalya.gatsh@omu.edu.ly* , Computer Science Department, Faculty of Arts and Sciences, Omar Al-Mukhtar University, Derna- Libya

advance. For example, real-world problems such as diagnosis and forecasting usually rely on an incomplete, uncertain, conflict or ambiguous knowledge **(**facts and rules**)** about the problem to make a suitable decision or conclusion (Negnevitsky, 2005). Consequently, solving an AI problem requires intelligence that primarily based on intuition (Konar, 2000). The main role of intuition is to filter and rearrange the acquired knowledge in a proper sequence to solve the problem at hand, where the reliability of an intuition-based decision is strongly influenced by the experience **(**trial and error**)** in the problem domain.

As an expected result of continued and hard efforts of AI scientists, the field of artificial intelligence has been enriched with variant techniques to solve AI problems (Negnevitsky, 2005; Russell & Norvig, 2010). The main inspirational resource of AI techniques is the human being. Whereas, most of AI scientists focus on *how a human can solve an intelligent problem?* Some of them concerned with the question, *how the human's brain works?* On this context, they have presented many successful techniques such as expert system and artificial neural network. However, genetic algorithm and simulated annealing were motivated by the natural phenomena as a secondary resource of AI techniques. From a practical standpoint, we can consider the proposed search strategies are an easier and more systematic way to solve AI problems without losing the ability to explain the solution. The term '**search**' generally refers to the process of finding a legal sequence of steps to solve a problem among all candidate configurations. While, the term '**solution**' stands for a path of intermediate states between the start state and the goal state, or an artifact state that satisfies some conditions (Russell & Norvig, 2010; Tim, 2008). Solving an AI problem by search requires reformulating it in the form of a search problem by deriving three main requirements (Russell & Norvig, 2010), and based on the nature of the given problem, they may be obvious or require deep insight to be recognized. *First*, specifying the start state that represents the initial status to solve the problem. In some cases, the start state is a part of the problem. However, if we have the choice to choose the start state of the search, we should select a start point from which the search can achieve the desired solution faster. *Second*, we have to determine the operators (actions) that we will apply to each state to yield new states. We also should select the operators of a problem carefully, because they are primarily responsible for expanding the search space that should consist of all the problem states that can lead to the target solution. *Finally*, identifying the goal test, which refers to *when the search process should terminate and present the results?* Usually, the search will terminate if it achieved the desired solution, or overcame a specified threshold even if the goal has not been found. After preparing an AI problem, we can now apply a search strategy to solve it. Our study aims to enhance and compare the performance of main search techniques based on the required time and memory to solve the same AI problem.

MATERIALS AND METHODS

During the journey of looking for a solution, each search strategy specifies a mechanism to discover the search space. Based on the applied mechanism, we can classify the AI search strategies to *exhaustive* and *heuristic* search (Chijindu, 2012). In general, exhaustive search strategies aim to solve an AI problem by exploring the whole search space a level-by-level or a branch-by-branch according to the sequence of chosen operators. The problem-solving procedure of exhaustive search strategies depends on the principle of '**compare and expand**'. In more details, it includes a comparison between each state in the problem search space and the predefined goal state. If there is no matching, the current state will expand by selecting the appropriate operators to produce legal states. Regardless of the generated states are new or previously visited many times, they will be stored to check them later. The process

of comparing and expanding will continue until the termination criterion is satisfied. Due to the comprehensive exploration of a problem space, exhaustive search strategies usually guarantee finding a solution to the problem, if there is one. On the other hand, some of them require much space of memory to store unvisited states or a very long time to reach the goal. Furthermore, exhaustive search strategies are blind in that they give all available states an equal chance to discover them, as well as they may leave the goal state behind due to their assertion to follow a deterministic arrangement to explore the search space. Figure1 (a) views the search space of a virtual problem in a tree representation that includes two paths to reach the goal state (G). The different ways of exhaustive search strategies to explore the proposed search space are presented in figures1 (b) and (c), where the numbers of labeled states refer to the order by which the search will visit the states according to each way. Whereas, unlabeled states denote to undiscovered states that still occupy space of the memory. As shown in figure1, a level-by-level search could find the shortest path (often is not the cheapest) to the goal with more stored states than a branch-by-branch search that traced a longer path to reach the desired solution after visiting more states.

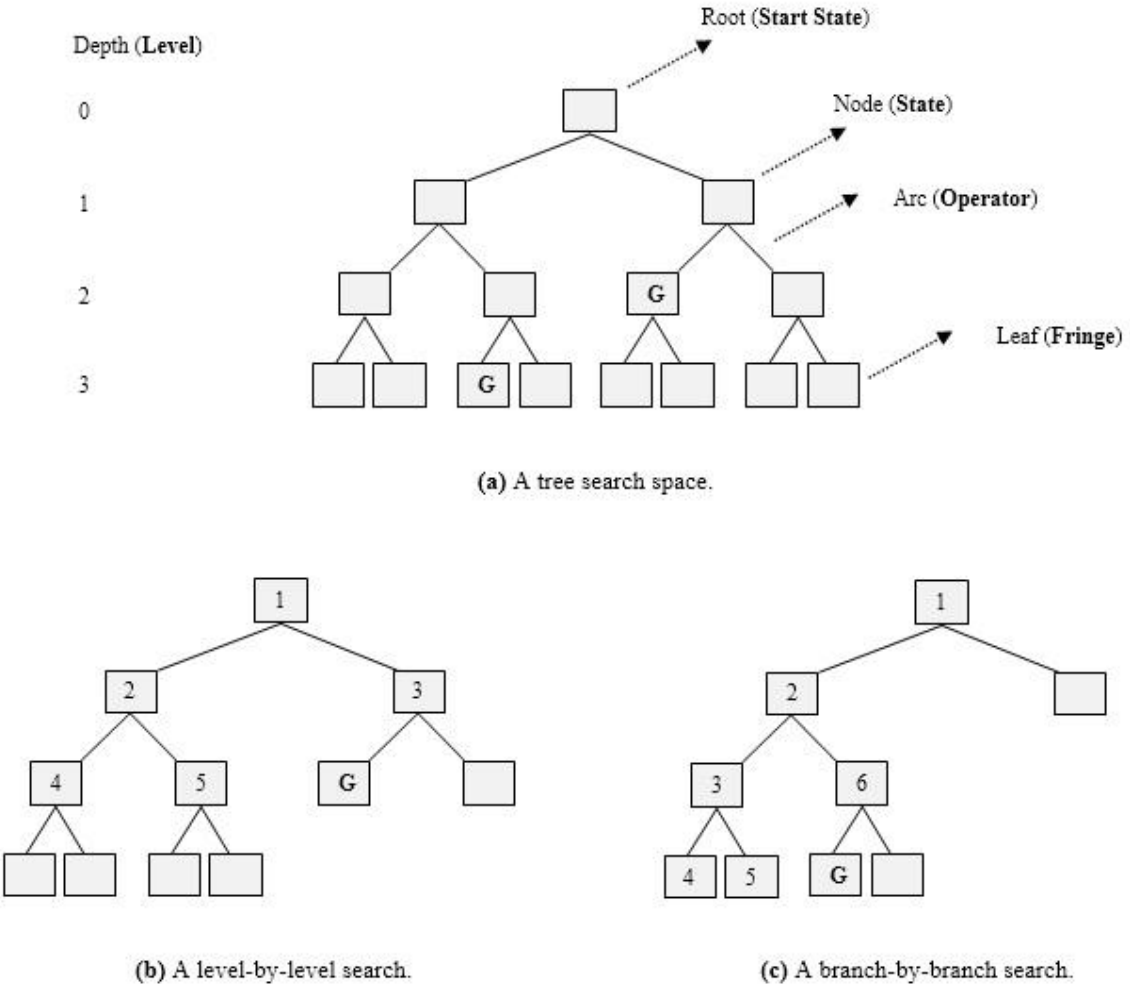


Figure1. The different ways of exhaustive search strategies for a virtual problem.

In contrast to exhaustive search strategies, heuristic search strategies rely on intelligent knowledge to efficiently prune the search space of a problem and focus on the most promising

branches for a solution. Consequently, when we decide to apply a heuristic search strategy to solve an AI problem, we should take into account a fourth preparing step that is specifying the appropriate heuristic function of the problem. The heuristic function represents a measure by which a heuristic search strategy can evaluate the quality of a state from the search space to be a suitable piece of a legal chain, which will finally lead the search to the goal state. Unfortunately, identifying the adequate heuristic function to guide the search professionally toward the solution is usually a difficult task, because it requires a deep understanding of the problem and human intelligence. While a carefully selected heuristic function can lead the search directly to find an optimal solution path with fewer visited states, an inefficient measurement will mislead the search toward unpromising branches that may require much effort to reach the goal than exhaustive search strategies due to the additional time of computations. Figure2 views an ideal performance of heuristic search strategies for the search space presented in figure1 (a). Where, the order of evaluated states and their scores, in comparative expressions, show above each state. However, the numbers within the virtual states denote to the arrangement of expanded states.

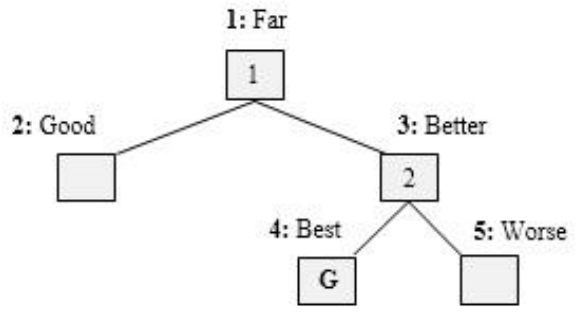


Figure2. An ideal performance of heuristic search strategies for the virtual problem.

As shown in figure2, the heuristic search strategies follow a nondeterministic order to discover the search space of the given problem based on the principle '**evaluate and expand**'. For each state in a problem search space, heuristic search strategies employ the proposed heuristic function to estimate the closeness of the current state from the goal. If the approximated distance is more than zero, the state will expand and its children are evaluated according to the same measurement. The process of evaluating and then expanding will be in progress until the search reaches the target state (a state that scores zero distance for its evaluation). Where, we can utilize this mechanism to look for the cheapest solution of a problem by inserting the exact cost of generating each state from the start state into the total evaluation of them (Poole & Mackworth, 2017). There are two manners of search to exploit the foreseen readings of a heuristic measure and decide the next state toward the solution. *First*, the local search **(**choose among children**)** that focuses on selecting the best newborn child of the current state to follow it. Although the local search has the ability to reach the target state faster (Russell & Norvig, 2010), if it starts with discovering the correct branch, the search may confuse when it faces some obstacles such as *a trap at local maxima* or *reaching a plateau* (Chijindu, 2012; Konar, 2000). As shown in figure3 (a), trap the search at local maxima means that the evaluation of a parent is better than its children scores. Whereas, reaching a plateau implies that the evaluation of a state and its offspring is identical, see figure3 (b). *Second*, the global search **(**choose among fringes**)** depends on a wide set of possibilities includes all unexpanded states in the search space to decide the nearest state to the goal. However, through solving a problem, the global search sometimes loses the path toward the goal state before tracing it again after exploring a number of redundant states (Chijindu, 2012; Russell & Norvig, 2010).

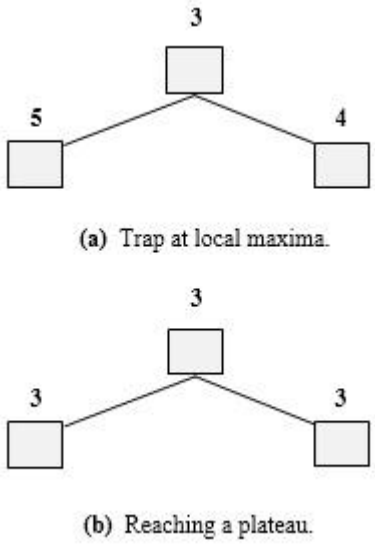


Figure3. The main obstacles of local heuristic search strategies.

RESULTS AND DISCUSSION

At this moment, we are going to illustrate the basic concepts of main AI search strategies based on a variant of the popular numbers-puzzle that known as the 4-puzzle game (Konar, 2000). As shown in figrue4, the 4-Puzzle game includes a squared board contains three cells labeled with the numbers 1, 2 and 3, where there is one place on the board left free **(**which here is distinguished by the letter B**)** to provide the ability to move the other cells. The point of this game is to rearrange the numbered cells that presented in the start state to match the goal state by moving them up, down, left, and right. The 4-Puzzle game is a simple example of an AI problem because there is no obvious algorithm to tell us each time, in advance, how we should move the cells to get the goal arrangement, so it depends on the player intuition.

As we mentioned early, we have to reformulate the problem before applying a search strategy to solve it. For the example of the 4-Puzzle problem that viewed in figure4, we already know the start point of the search and the form of the desired solution is the sequence of applied movements to reach the specified goal state. To gradually generate the search space, we can use a wide set of operators includes twelve actions that represent the four available movements for each numbered cell. However, this will require much time to examine the ability of applying each operator on the current state to produce a new one. Also, using a wide set of operators may cause a huge search space that needs much memory space. On the other hand, if we based on a very small set of operators, it may reduce the chance of achieving the target state or increase the efforts to reach the solution. Therefore, we should carefully choose an adequate set of operators to guarantee solving the problem as soon as possible with fewer memory requirements. Consequently, we will rely on four operators to rearrange the 4-Puzzle cells, which are move Blank-Up **(**BU**)**, move Blank-Down **(**BD**)**, move Blank-Right **(**BR**)**, and move Blank-Left **(**BL). Actually, we cannot move the blank space, but we can replace its position with one of the cells around it. Generally, the search terminates when it reaches the goal state. However, we should have an auxiliary condition that usually associated with the applied strategy to interrupt the search even though it has not found the goal state yet. In a level-by-level exhaustive search or the global heuristic search, the search process will terminate if there is no new region to discover it. This condition is applicable, if and only if, we supported the search with an additional procedure to filter new states among the old ones. Whereas, the search will interrupt if we apply a branch-by-branch exhaustive search when it explores all search space branches until a specified max depth. Although the max depth is a common termination criterion, it usually requires deep insight to specify it. Because choosing a big depth to explore each branch will maximize the chance of finding the desired state but minimize the probability to reach a short path to the solution, and vice versa if we decided to discover a small fraction of the search space. Finally, the auxiliary termination condition of the local heuristic search is that stop visiting new states of the search space if

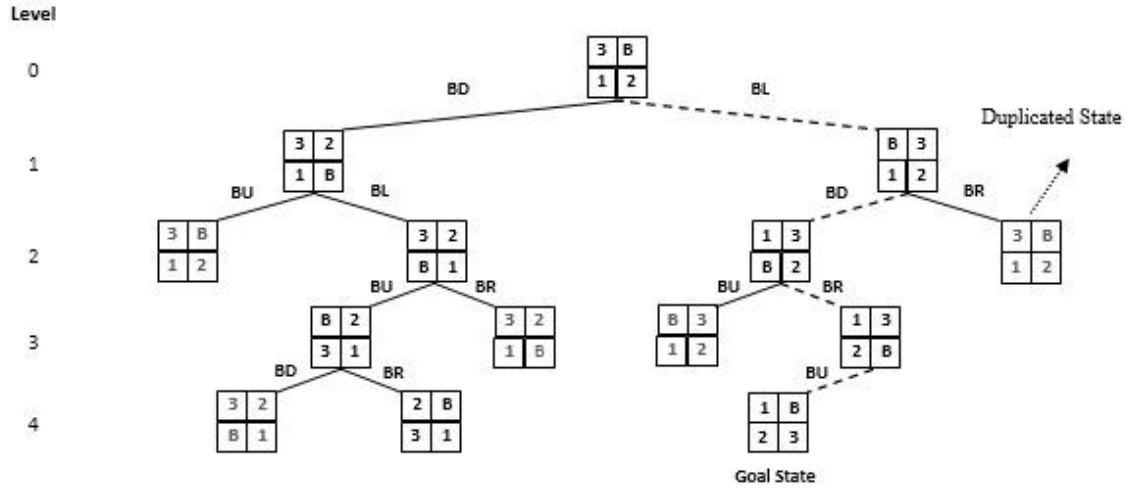
there is no improvement in the evaluation results of generated states.

Figure4. An example of the 4-Puzzle problem.

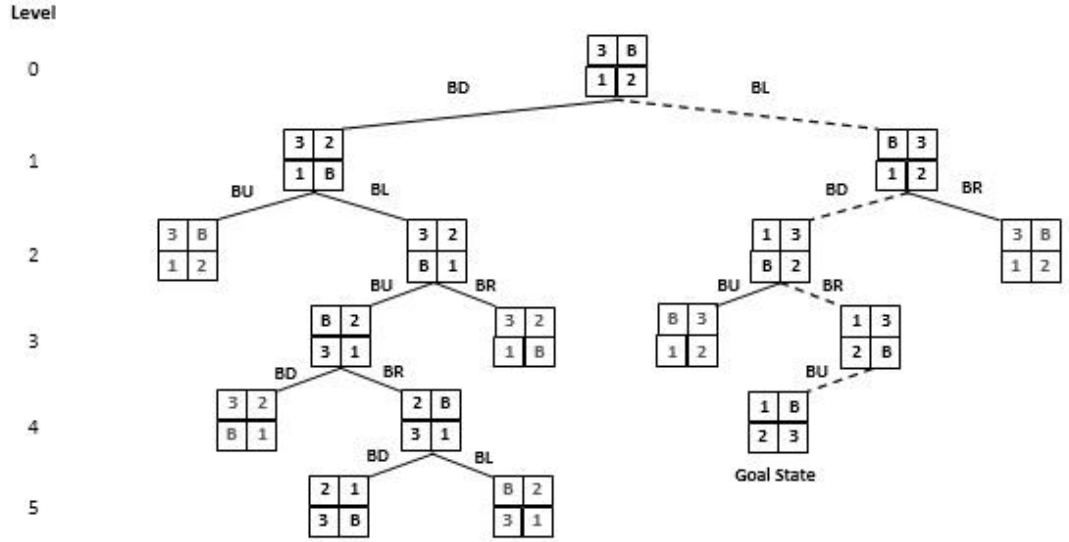
After preparing the 4-Puzzle problem for search, we can now clarify the performance of different search strategies to solve it. First, we are going to apply the variant techniques of exhaustive search strategies to solve the problem. However, in the beginning, we will introduce some suggestions to improve the behavior of these techniques (Russell & Norvig, 2010; Tim, 2008). As we previously stated, exhaustive search strategies do not distinguish between new and old states during solving an AI problem. Where, accepting duplicated states is usually responsible for the explosion growing of the search space, which then requires a lot of time and memory to find a solution. As well as, it may trap the search at an infinite loop of repeated states that prevents the ability of a branch-by-branch search to reach the goal. On the other hand, if the search can focus only on visiting unique states and using an additional procedure, it will achieve the solution faster. Although identifying repeated states requires keeping in memory all states that previously expanded or those waiting to expand later, surprisingly, it usually needs a less memory space than visiting old states many times. keeping in mind that the role of eliminating previously visited states will decrease or be useless if the number of duplicated states is very small compared to new ones. The second proposition to enhance the performance of exhaustive search is that we can make the search more conscious or not blind if we force it to compare each recently generated state to the goal state before storing it in the memory, in case there is n Store the start state in the waiting list match between them. Consequently, an exhaustive search strategy will terminate the process of discovering the search space directly if it realized the goal state without any delay because it follows a specific order to solve the problem. Based on these suggestions, we are going to apply the basic algorithm of exhaustive search strategies that presented below to find a solution path of the 4-Puzzle problem.

Keep in mind that storing and retrieving a state from the waiting list must serve the search mechanism of the applied strategy. Figure5 (a) and (b) views the performance of both exhaustive search strategies, a level-by-level and a branch-by-branch respectively, where we distinguish the solution path by a disconnected line between the start state and the goal state. Beside each arc, we presented the applied operator to generate a new state from the parent state. For a level-by-level exhaustive search, the search process will stop if it found the desired solution or there is no new region to discover. As expected, a level-by-level search traced the shortest path to the goal, where the search recognized the target state at the fourth level during expanding the states of the third level. Due to avoiding duplicated states, the search rapidly achieved the solution after expanding seven states while only a state is waiting in memory. In a branch-by-branch exhaustive search, the search will continue exploring each branch five levels (new states at the max depth will not expand) to give a chance for other branches to discover them, so we need to update the above algorithm slightly to ensure that. Moreover, when we support a branch-by-branch search with a procedure to prevent repetitive states, the search will discard only the state that previously visited at a higher level. As shown in figure5 (b), although the applied strategy also achieved the shortest path to the goal, the search spent much effort to realize it where the strategy expanded eight states to reach the goal state without any waited states.

Do
Retrieve a state from the waiting list and call it the current state
Do
Apply an operator to the current state to generate another state
If (the generated state is the goal state**) then**
present the discovered solution and interrupt the search
If (the generated state is new**) then**
Store the generated state in the waiting list
While (there is a new operator to apply to the current state**)**
Store the current state in the visited list
While (there is a state in the waiting list**)**



(a) A level-by-level search.



(b) A branch-by-branch search.

Figure 5. Solve the 4-Puzzle problem by the enhanced exhaustive search strategies.

Figure 5 showed a solving example of the 4-Puzzle problem by different exhaustive search strategies that successfully guided the search without any assistance knowledge. Suppose we can provide the search process with simple knowledge to evaluate each state in the search space based on the principle, *how match the current state the goal state*? In other words, we can use a heuristic search strategy to solve the same example of the 4-Puzzle based on a simple heuristic function that evaluates the goodness of a state by counting the number of mismatched cells comparing to the goal state. We are going first to apply the local heuristic search, where the primary termination criterion will be finding out a solution whereas the secondary one is interrupting the search if all search space's branches expanded ten levels without any improvement. During the search, if there are many candidate children to follow **(**all of them have the least distance to the goal state**)**, the search will choose the first generated state according to the arrangement of applied operators. Furthermore, since our simple measure is unable to provide the search with an accurate reading of a state closeness from the goal, it may give repeated states a chance to expand them again, therefore we need a supplementary procedure to explore each state once as described in the next algorithm. Figure6 displays how the proposed heuristic function led the search toward the solution, while the numbers upper of states denote to their evaluations. As you see, the search kept visiting states with the same score for five levels, after that, the states evaluations continued improving until reaching the goal state at the eighth level. We conclude that the imprecise evaluations and the mechanism of selection together misled the search toward a longer solution path that required more computational time in contrast to exhaustive search strategies.

Store the start state in the waiting list after evaluating it
Do
    Retrieve the closest state to the goal from the waiting list and call it the current state
    If (the current state is the goal state**) then**
        present the discovered solution and interrupt the search
    Do
        Apply an operator to the current state to generate another state
        If (the generated state is new**) then**
            Evaluate its closeness from the goal state
    While (there is a new operator to apply to the current state**)**
    If (the current branch is not useless**) then**
        Store new generated states in the waiting list in ascending order based on their evaluations
        Store the current state in the visited list
While (there is a state in the waiting list**)**

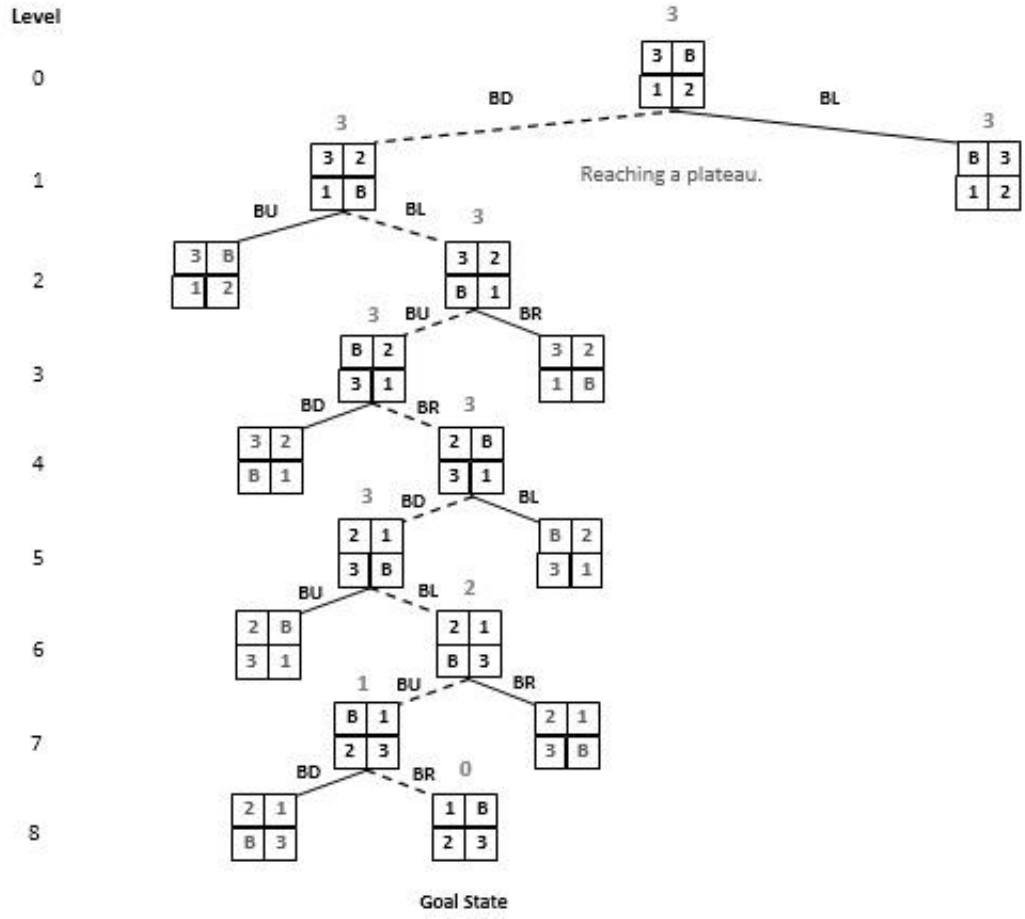


Figure 6. Solving the 4-Puzzle problem based on local heuristic search.

The algorithm presented above needs a little change to be suitable for the global heuristic search too, which is saving the new generated states unconditionally in the waiting list. Consequently, the global search will stop if either it found a solution or the waiting list is empty. Where, if the search faced occasionally a number of fringes with identical evaluations to choose among them the next state, it will select the oldest state in the memory. As shown in figure7, the global heuristic search presented a better performance than local heuristic search to solve the 4-Puzzle problem based on our simple heuristic function. At the beginning, the search missed the solution path to discover a redundant state before correcting its trajectory toward the goal state directly. Although the additional efforts of evaluations, the global heuristic search presented a better performance than the adjusted level-by-level blind search

On the other hand, if we rely on proper knowledge to trace the best choice among available children, a heuristic search will realize a concise path to the solution without expanding any redundant states. Therefore, we are going now to solve the 4-Puzzle problem with an advanced heuristic function that uses an alternative principle to precisely evaluate the search space which is, *how far is the current state from the goal state?* As described in the algorithm below, the heuristic function will compute the total evaluation of a state based on how each of its cells is far from its equivalent cell in the target state. In this context, the evaluation of the start state that presented in figure4, clockwise, starting from the cell at the upper left is (2+1+1=4). As we expected, the

new heuristic function perfectly led the search toward the desired state, review figure8.

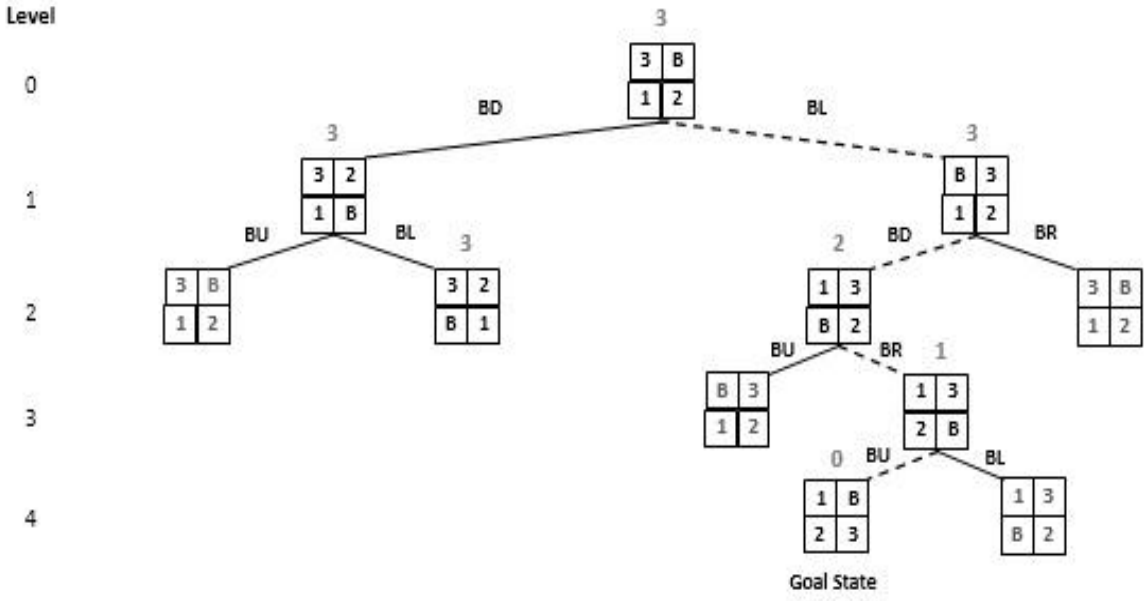


Figure7. Solving the 4-Puzzle problem based on global heuristic search.

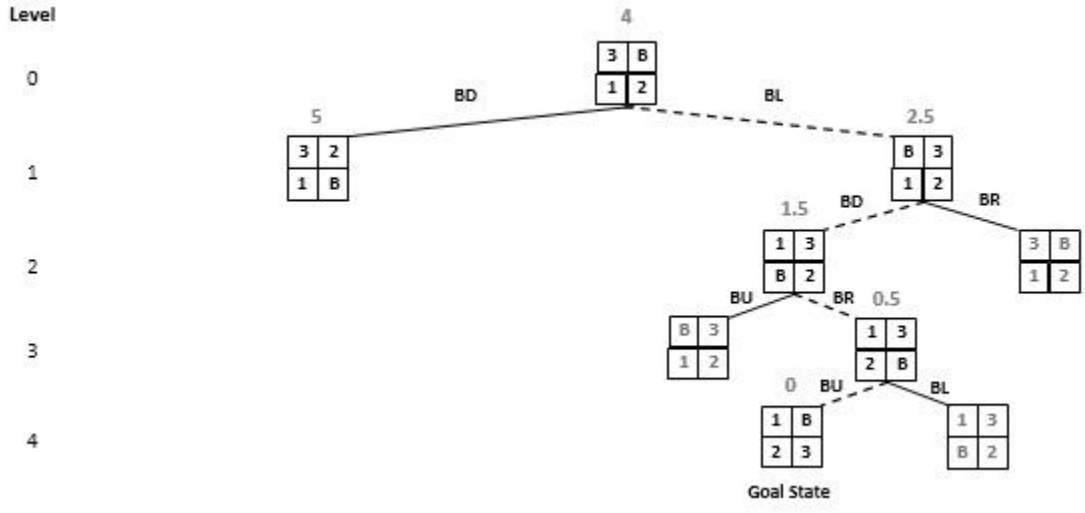


Figure8. Solving the 4-Puzzle problem based on advanced heuristic function.

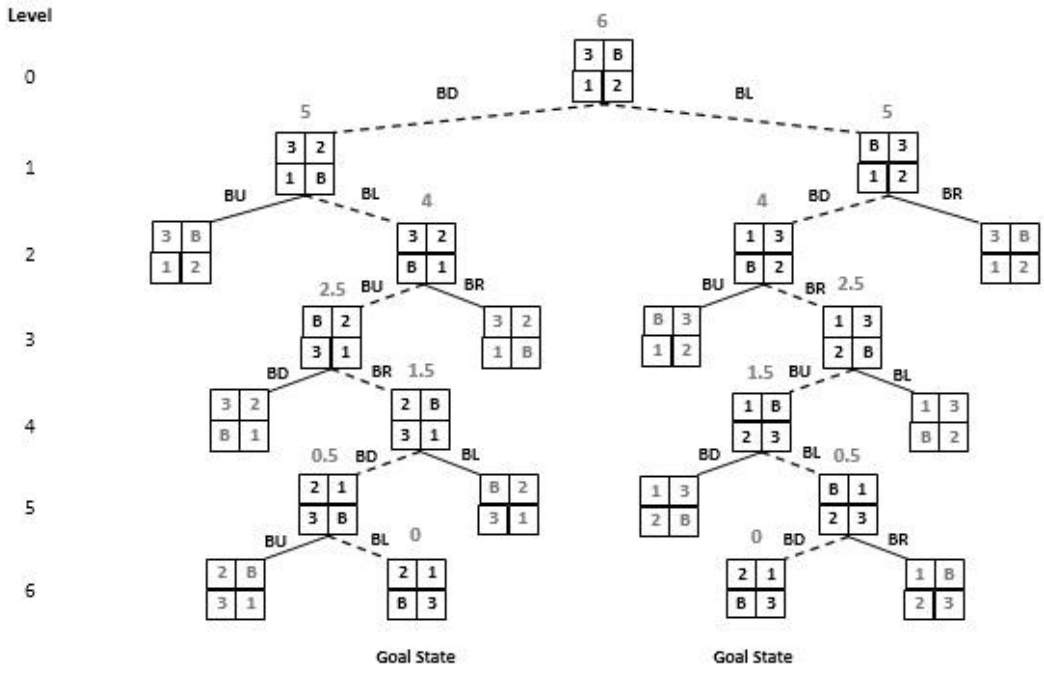
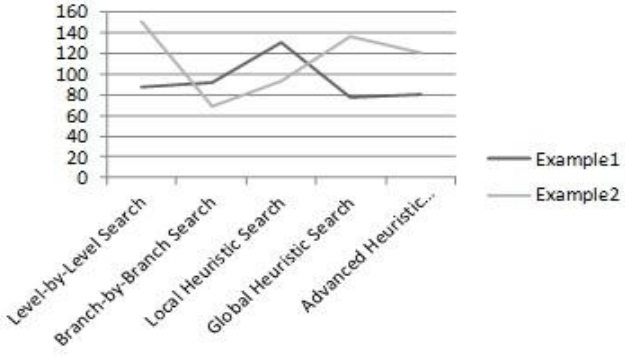
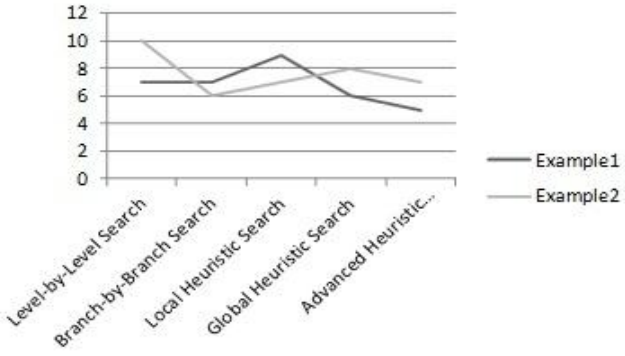


Figure9. Solving another example of 4-Puzzle using the advanced heuristic function.



(a) The required time to solve each example.



(b) The required memory to solve each example.

Figure10. Comparing the performance of all search strategies over two examples of the 4-Puzzle problem.

Because of our new measure always guides the search to explore new states that have better evaluations than previously discovered ones, we do not expect to trap the search at local maxima. Moreover, if the search got a number of newborn children that have the best evaluation, this means there are many paths to reach the goal state at the same level so we can choose any one of them at random. As demonstrated in figure9, the proposed heuristic function could provide the search with accurate readings about the search space for a new example of the 4-Puzzle problem where we decided to explore both branches at the first level to examine the quality of the applied fitness measurement. Even though our advanced measurement does not visit a state twice, we will accomplish the last two experiments with discarding repeated states to reduce the evaluation efforts. Where, the significance of avoiding duplications increases when the evaluation process is very complicated and requires a long time (Poole & Mackworth, 2017). Unfortunately, the applied strategy spent slightly much time than the global search to reach the goal state in the first example of the 4-Puzzle, but its performance was better than other search strategies that we examined earlier. To ensure that our results are more reliable, we decided to apply all search strategies again to the example that is presented in figure9. Figure10 showed a comparison of all search mechanisms over the two 4-Puzzle examples based on the required time and memory to solve them.

As shown in figure 10(a), a branch-by-branch exhaustive search had the best performance in the second example because of it started with exploring the shortest solution path. As well as, for the same reason, the local heuristic search presented a better performance than the first example. On the other hand, the level-by-level exhaustive search showed the worst behavior due to its need to look for the solution deeply. Unfortunately, the global heuristic search had a worse performance than the first example because there were many solution paths to trace. Finally, the advanced heuristic search showed a better performance than global and level-by-level searches because it was based on an accurate measure to guide the search directly toward the solution. However, because of the evaluation efforts, its performance was worse than local and branch-by-branch searches. Figure 10(b) demonstrated that avoiding duplicated states decreases the number of stored states where the branch-by-branch search had a little better chance to store fewer numbers of states than the advanced heuristic search if it discovered the adequate path first.

CONCLUSION

In this paper, we discussed the performance of different search techniques to solve two examples of a simple artificial intelligence problem. However, deciding which strategy is an adequate to solve an AI problem relies on two important factors: your ability to provide the search with an assistant knowledge about the given problem (simple or complex), and the main characteristic of the desired solution (the shortest, the fastest or the least memory). Although exhaustive search strategies, in general, guarantee finding an adequate solution for a problem without any a prior knowledge except its specification, they are usually time-consuming and memory wasting strategies. Alternatively, heuristic search strategies are more efficient in guiding the search toward the desired solution based on an intelligent knowledge in the form of a heuristic function that receives a state and returns its evaluation. Unfortunately, inexact knowledge may get the search away from the target path, whereas precise knowledge requires much time to solve the given problem.

REFERENCES

Chijindu, E. V. (2012). Search in Artificial Intelligence Problem Solving. *www. ajocict. net, 5*(5), 37 .

Konar, A. (2000). *Artificial intelligence and soft computing: behavioral and cognitive modeling of the human brain*: CRC press.

Negnevitsky, M. (2005) .*Artificial intelligence: a guide to intelligent systems*: Pearson education.

Poole, D. L., & Mackworth, A. K. (2017). *Artificial Intelligence: foundations of computational agents*: Cambridge University Press.

Russell, S. J., & Norvig, P. (2010). Artificial Intelligence-A Modern Approach (3rd internat. edn.): Pearson Education.

Tim, J. M. (2008). Artificial Intelligence–A System Approach. *Computer Science Series, Infinity Science Press, 498* .

مجلة المختار للعلوم 34 (2): 91-104، 2019

الاختيار ما بين البحث الشامل أو الاجتهادي لحل مشاكل الذكاء الاصطناعي

داليا أبوبكر قاطش

قسم الحاسوب، كلية الآداب والعلوم، جامعة عمر المختار، درنة–ليبيا.

تاريخ الاستلام: 02 أبريل 2019 / تاريخ القبول: 05 سبتمبر 2019

Doi:

المستخلص: يعد الذكاء الاصطناعي مجال بحث مثير للاهتمام في علم الحاسبات؛ وذلك لأنه يركز على اكتشاف تقنيات فعالة تم استيراد فكرتها بشكل رئيس من الإنسان أو بيئته المعيشية لحل مشاكل ذات طبيعة خاصة. في هذا البحث، سنسعى أولاً إلى تقديم توضيح مفصل للخصائص المشتركة للمشاكل التي يهتم بدراستها مجال الذكاء الاصطناعي، من ثم سنقوم بتسليط الضوء على كيفية تهيئة مثل هذه المشاكل لحلّها من خلال البحث. الهدف الرئيس من الدراسة هو مساعدتنا في تحديد إستراتيجية البحث الأفضل من خلال دراسة سلوك إستراتيجيات البحث الأكثر شيوعاً لاكتشاف الحل المرغوب لمثالين من مشكلة ذكاء اصطناعي بسيطة. نتائج تجاربنا أظهرت أن الوقت وسعة الذاكرة اللازمة لحل المشكلة يتأثران بعدة عوامل منها آلية البحث، موقع الحل، عدد الحلول المتاحة والتعقيد في البحث.

الكلمات المفتاحية: مشكلة ذكاء اصطناعي، مشكلة بحث، البحث الشامل، البحث الاجتهادي.

*داليا أبوبكر قاطش dalya.gatsh@omu.edu.ly قسم الحاسوب، كلية الآداب والعلوم، جامعة عمر المختار، درنة–ليبيا.