1-1-2020

Performance Evaluation of Pathfinding Algorithms

Harinder Kaur Sidhu
University of Windsor

Performance Evaluation of Pathfinding Algorithms

By

Harinder Kaur Sidhu

A Thesis
Submitted to the Faculty of Graduate Studies
through the School of Computer Science
in Partial Fulfillment of the Requirements for
the Degree of Master of Science
at the University of Windsor

Windsor, Ontario, Canada

2019

Performance Evaluation of Pathfinding Algorithms

by

Harinder Kaur Sidhu

APPROVED BY:

_____

C. Semeniuk

Great Lakes Institute for Environmental Research

_____

A. Jaekel

School of Computer Science

_____

S. Goodwin, Advisor

School of Computer Science

December 4, 2019

DECLARATION OF ORIGINALITY

I hereby certify that I am the sole author of this thesis and that no part of this thesis has been published or submitted for publication.

I certify that, to the best of my knowledge, my thesis does not infringe upon anyone's copyright nor violate any proprietary rights and that any ideas, techniques, quotations, or any other material from the work of other people included in my thesis, published or otherwise, are fully acknowledged in accordance with the standard referencing practices. Furthermore, to the extent that I have included copyrighted material that surpasses the bounds of fair dealing within the meaning of the Canada Copyright Act, I certify that I have obtained a written permission from the copyright owner(s) to include such material(s) in my thesis and have included copies of such copyright clearances to my appendix.

I declare that this is a true copy of my thesis, including any final revisions, as approved by my thesis committee and the Graduate Studies office, and that this thesis has not been submitted for a higher degree to any other University or Institution.

ABSTRACT

Pathfinding is the search for an optimal path from a start location to a goal location in a given environment. In Artificial Intelligence pathfinding algorithms are typically designed as a kind of graph search. These algorithms are applicable in a wide variety of applications such as computer games, robotics, networks, and navigation systems. The performance of these algorithms is affected by several factors such as the problem size, path length, the number and distribution of obstacles, data structures and heuristics. When new pathfinding algorithms are proposed in the literature, their performance is often investigated empirically (if at all). Proper experimental design and analysis is crucial to provide an informative and non-misleading evaluation. In this research, we survey many papers and classify them according to their methodology, experimental design, and analytical techniques. We identify some weaknesses in these areas that are all too frequently found in reported approaches. We first found the pitfalls in pathfinding research and then provide solutions by creating example problems. Our research shows that spurious effects, control conditions and sampling bias data can provide misleading results and our case studies provide solutions to avoid these pitfalls.

DEDICATION

I dedicate this work to my fiancée Mandeep Singh.

ACKNOWLEDGEMENTS

I would like to express my sincere gratitude towards my advisor Dr. Goodwin for guiding me through my thesis and also for being patient with me and understanding the whole time. I would like to thank him for providing me the knowledge and resources like books and papers required to complete my research work and providing continuous feedback on my work to make it better.

I would also like to thank my committee members Dr. Jaekel and Dr. Semeniuk for their support and time.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF APPENDICES

CHAPTER 1

Introduction

1.1 Thesis Claim

In this thesis, we proposed the set of guidelines for the researcher to design the experiment and to empirically analyze the results from the data collected. In pathfinding when a new algorithm or data structure is proposed it is tested on set of different maps. Therefore, the main focus of this thesis is to study the various aspects of maps, generated in pathfinding experiments, like the path length, the density of obstacles, map size and so on as the performance of newly built algorithm is tested on these maps thus it is an important aspect of pathfinding experiments. We studied over 150 research papers related to pathfinding and found that nearly 65% of papers do not provide enough information to support their claim and nearly 50% of these papers have only either random grid maps or game maps to test their algorithm. Only 20% of the papers conduct sound experiments and provide empirical analysis of their results. While reviewing these papers we found some weaknesses in the experimental setup based on the design of the experiments, methodology and analytical techniques used by researchers. Our thesis shows that it is really important to avoid these pitfalls and design the experiments empirically sound to test the algorithm for all possible outcomes to do a detailed analysis of the algorithms' performance. Based on our findings, we categorized the maps into four types to consider different obstacle distributions of obstacles and also show how different attributes of maps can be manipulated to get more reliable results. We support our findings with some case studies which shows that deviating from these practices give unreliable and misleading results.

1.2 Pathfinding- Overview

Pathfinding refers to computing an optimal route in a given map between the specified start and goal nodes. It is an important research topic in the area of Artificial Intelligence with applications in fields such as GPS, Real-Time Strategy Games, Robotics, logistics while implemented in static or dynamic or real-world scenarios [1]. Recent developments in pathfinding lead to more improved, accurate and faster methods and still captivates the researcher's attention for further improvement and developing new methods as more complex problems arise or being developed in AI [2]. A great deal of research work is done in pathfinding for generating new algorithms that are fast and provide optimal path since the publication of the Dijkstra algorithm in 1959. Most of the research work is validated using experimental data. Therefore, the research must provide reliable and accurate information as experiments are very volatile.

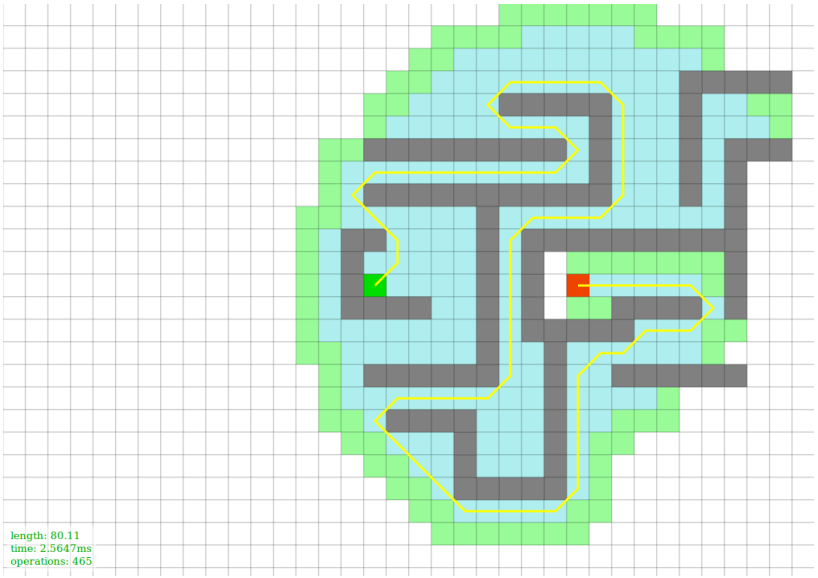


Figure 1: Pathfinding Example Map

1.2.1 Pathfinding Problem Definition:

Pathfinding is closely related to shortest path problem, thus the definition of pathfinding is finding the optimal path from a given start node(s) to goal node(g) in the given graph(G), where optimal refers to the shortest path, low-cost path, fastest path or any other given criteria. Pathfinding can generally be divided into two categories: SAPF, that is Single Agent Pathfinding, to generate a path for one agent and MAPF, that is Multi-Agent Pathfinding, to generate the path for more than one agent. In this paper, we only consider the single-agent pathfinding problem in a static environment, which means the map does not change as the agent moves. Pathfinding has applications in different fields and it is hard to consider all the application areas, so in this paper, only video game applications are used and in 2D environments.

1.3 Map Representations:

Pathfinding is used in a wide variety of areas and usually implemented on different maps that are generated to test pathfinding algorithms. The widely popular maps are implemented using a grid-based graph, set of nodes and edges, representations in the algorithm. Usually, a grid is superimposed over a map and then the graph is used to find the optimal path [4]. Most widely used representations are square tile grid which can either be accessed as a 4-way path or 8- way path. Both have their own advantages and disadvantages. Grids are considered simple and easy to implement and are commonly used by researchers. Other representations are Hexagonal grid, Triangular grid, Navigation Mesh, and Waypoints [1].

Various types of map representations are discussed below briefly:

1.3.1 Tile Grids:

The composition of the grid includes vertices or points that are connected by edges. Basically, grids uniformly divide the map world into smaller groups of regular shapes called “tiles”. The movement in square tile grids (Fig 2.a) can either be 4-way (no

diagonal movement) or 8-way (diagonal movement). The second most widely used grid representations are Hexagonal grids (Fig 2.b). Hexagonal grids are like square grids with the same properties and take less search time and reduced memory complexities [5]. Triangular grids (Fig 2.c) are not popular among game developers and researchers but some methods are proposed to reduce the search effort and time consumption.

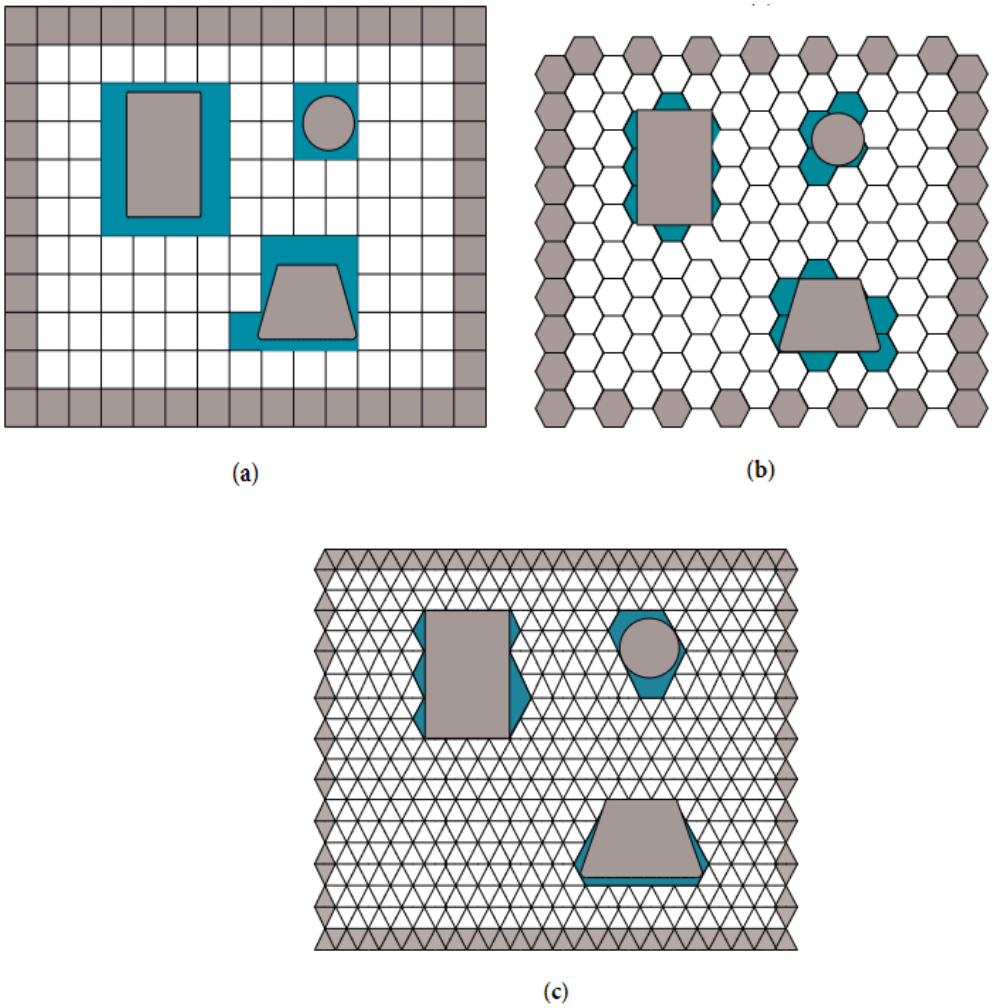


Figure 2: Different types of tile grids[19]

1.3.2 Navigation Mesh:

A navigation mesh (Fig 3) is a connected graph of convex polygons, where the polygon is a node in a graph, also known as navmesh. Polygons represent a walkable area, thus movement in any direction is possible within the polygon. The map is pre-processed to generate nav-mesh and then the path can be found by traversing polygons (from polygon consisting start point to polygon consisting goal point). The benefits of using navigation mesh are that it reduces the number of nodes in the graph as the large walkable area can be represented as a single convex polygon, reduces the memory required to store pre-processed map, and increases the speed of pathfinding [6].

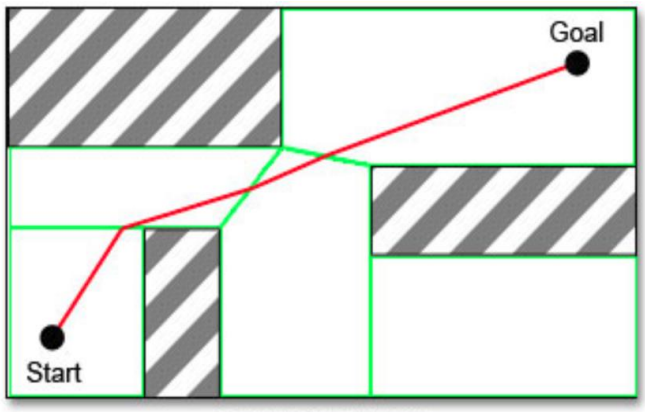


Figure 3: Navigation mesh[18]

1.3.3 Waypoints:

A waypoint (Fig 4) can be defined as a point along the path which can be marked manually or can be automatically computed. The purpose of waypoints is to minimize the path representation as the shortest path can be pre-computed between any two points. Therefore, certain optimization techniques are developed to compute the path using waypoints. The main advantage of waypoints can be in a static world as the map does not change, so the shortest paths between two waypoints can be pre-computed and stored, reducing the time to calculate the final path after execution.

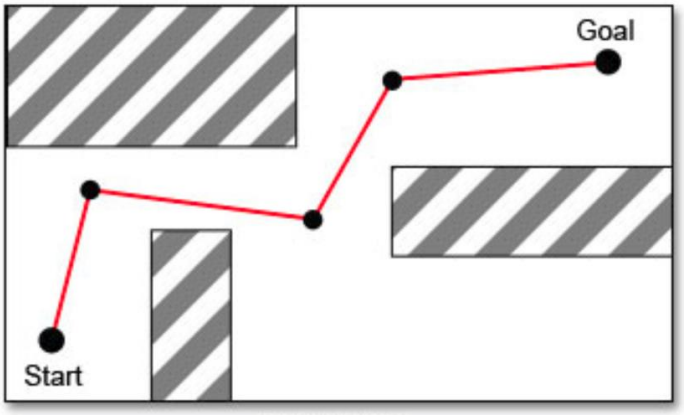


Figure 4: Waypoints[18]

1.4 Algorithms

For finding a path between two nodes in a given graph a search algorithm is required. Many search algorithms have been developed for graph-based pathfinding. Pathfinding algorithm generally finds the path by expanding nodes and neighboring nodes according to some given criteria. Pathfinding algorithms can be broadly divided into two categories: Informed and Uninformed pathfinding Algorithms.

1.4.1 Informed Pathfinding Algorithms:

As the name suggests informed means having prior information about the problem space before searching it. Informed search refers to the use of knowledge about the search space like problem map, estimated costs, an estimate of goal location. Thus, the algorithm utilizes this information while searching a path and it makes pathfinding fast, optimal and reduces memory usage in node expansion [7]. Various algorithms that fall under this category are A*, IDA*, D*, HPA*, and many more. These algorithms use different heuristic functions or uniform cost function to utilize the

information of the search problem. The following heuristic functions, used by these algorithms, are discussed briefly here:

Manhattan Distance: Manhattan distance is considered as a standard heuristic for the square grid, defined as the sum of the absolute difference between the start and goal position cartesian co-ordinates. In pathfinding, the Manhattan distance is the distance between start node to goal node when the movement is restricted to either vertical or horizontal axes in a square grid. The heuristic function is given below:

$$h(x) = |x1 - x2| + |y1 - y2|$$

Octile Distance: Octile distance is the distance between two points when diagonal movement is possible along with horizontal and vertical. The Manhattan distance for going 3 up and then 3 right will be 6 units whereas only 3 units diagonally (octile distance). The function of octile distance is given below:

$$h(x) = \max((x1 - x2), (y1 - y2) + (\text{sqrt}(2) -1) * \min((x1 - x2), (y1 - y2))$$

Euclidean Distance: When any angle movement, not the grid directions (horizontal. vertical, diagonal), is allowed then the straight-line distance is the shortest distance between any two points which is also known as Euclidean distance. The function is given below:

$$h(x) = \text{sqrt} (|x1 - x2| + |y1 - y2|)$$

In uniform cost search the next node is selected based on the cost so far, so the lowest cost node gets selected at each step. It is complete and optimal but not efficient as it takes lot of time to explore nodes.

1.4.2 Uninformed Pathfinding Algorithms:

Uninformed pathfinding refers to finding the path without any knowledge of the destination in the search space with only information about start node and adjacent nodes, also known as blind search [7]. Thus, the algorithm blindly searches the space by exploring adjacent nodes to the current node. Breadth-first search, depth-first search, Dijkstra are some algorithms that fall under this category. Uninformed search is slow and consumes lots of memory in storing nodes as it searches whole space until the destination node is found. The uninformed pathfinding is also known as an undirected search approach, which simply does not spend any time in planning. It just explores the nodes that are connected with the current node and then explore their neighbor nodes and so on until finds the node marked as goal node.

1.5 A Algorithm and its variants:*

The A* algorithm is popular among all fields of pathfinding. A* algorithm was first proposed in 1968 by Hart et al. and then improved version in 1972. A* algorithm combines the actual cost from the start point and estimated the cost to the endpoint to choose the next node to be explored. The estimated cost is given by the heuristic function used in A*. Given a start node in a graph, A* always finds the optimal path to the goal node [7]. The process involves building all possible paths from the starting node and exploring the adjacent nodes one at a time until reaches the node-set as a destination node. A* uses the "f" value given as:

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

where g(n) is the distance between the start node to some node n, h(n) is the estimated cost by heuristic function from node n to the goal node. A* uses the sum of these values to choose the next node to be explored in each step. A* is guaranteed to find the optimal path, given that the heuristic function must be admissible, by admissibility it means that the heuristic function never overestimates the cost than

the actual cost to reach the goal. The map is preprocessed, and all the information is stored as nodes and related costs are assigned to the nodes. A* maintains two lists while searching a problem space, one is Open List, and another is Closed List. The Open List consists of all the nodes that are not expanded yet but visited by the algorithm and the Closed List consist of all the nodes that are completely expanded means all the linked nodes have been visited by the algorithm. When the goal node is visited the algorithm terminates and then backtracks the nodes from the closed list to generate the path.

As A* is widely used and due to its popularity for finding the optimal path, many variants have been developed over the years to improve the performance and efficiency of the algorithm. The variants of A* are like Iterative-Deepening A*, Hierarchical Pathfinding A*, D*, Lifelong Planning A*, and so on.

1.6 Thesis Contribution

In pathfinding, when a new algorithm or any new improvement to the existing method is proposed, the results are validated using experimental results. These experiments are designed or meant to explore the reasons for algorithm performance and to confirm the findings through results collected from these experiments. Although experiments are very crucial for empirical studies these are also a very volatile and unstable sources of getting or validating results. In pathfinding, various maps and their representations are used for experimental setup to test the proposed algorithm or data structure. Our thesis focuses on exploiting these maps and their features to help design more sound experiments. Then we provide some general guidelines for conducting empirical research and writing a research paper in pathfinding. Our next focus in the thesis is to highlight various pitfalls in designing experiments, collecting results, analysis techniques used to extract important information from these results and representing the results in research papers.

When we study the literature work in pathfinding we realized that most of the papers lack in providing some important information in their research paper. We then critically reviewed the research papers and based on the findings we provide solutions to the common pitfalls through case studies that followed the proposed solutions. We broadly classified maps into four types: Random maps, Terrain or real-world maps, Floor plan or building plan maps and lastly game or maze maps. These four categories of maps cover almost every possible distribution of obstacles in the map one can create or might use when implementing algorithms. Most of the papers in pathfinding are either using random maps or game maps thus in some way getting a biased sample of data for their result analysis which as we know does not provide reliable information about the performance of their proposed method. Our thesis recommends future researchers in pathfinding to use at least these four types of maps to evaluate the results for their given method or algorithm.

1.7 Thesis Organization

We organized this thesis into six chapters. The first chapter introduces the pathfinding problem, its components, and underlying concepts. This chapter first gives the definition of pathfinding problem then briefly discusses the three main map representations: Tile grids, Navigation mesh, and Waypoints. After that, algorithms used in pathfinding are discussed, with major focus on A*, its variants and finally the contribution of this thesis. The second chapter gives the introduction to empirical research and its need for pathfinding research. Then we discussed the problems with existing research papers based on literature review and criticized some papers published in this field. The third chapter constitutes the motivation for our thesis and then gives details about empirical research and its basic components. Some general guidelines for conducting an empirical study in pathfinding are mentioned in this chapter. The fourth chapter highlights common pitfalls in pathfinding experiments

and provides solutions through case studies of problems that show the effects of pitfalls. The experimental setup used for the case studies is mentioned in this chapter. Chapter five concludes our thesis findings and recommended methods for conducting experiments. The final chapter is about possible future work to be done to extend our thesis.

CHAPTER 2

Introduction to empirical research and literature review

2.1 Empirical Research

Empirical research is an important aspect of research in the field of pathfinding. When a new pathfinding algorithm is proposed in the literature, the performance is evaluated by empirical methods. Empirical methods are a combination of exploratory techniques and confirmatory procedures. Exploratory techniques are those techniques that provide visualization, summarization, and modeling of the data collected by confirmatory experiments. Empirical methods amplify our observations and help us understand the structure of our problem world. Empirical studies seek the explanation for the performance of the algorithm rather than finding the best performing algorithm. In pathfinding, many works of literatures published were not designed or evaluated empirically. There are no set guidelines for conducting experiments in pathfinding. It is crucial to have a standardized experimental setup to evaluate the performance of the algorithm empirically, to conclude reliable results because whatever we publish presently becomes the fundamentals of the future. Therefore, if a slight weakness or not reliable information gets into the mainstream it will lead to more chaos in the future. In the literature, we did not find any paper which could provide some standards or guidelines to conduct empirical studies in pathfinding. Empirical research thoroughly examines the performance and provides experimental verification of the working of the method.

The advancement in artificial intelligence in games and other fields is making the pathfinding problem more challenging as the resources are utilized in other AI

operations like graphics, player actions, leaving a very small space for running pathfinding search. Also, there is a pressure of developing a more advanced, fast and optimal path planning search with limited resource utilization and minimum time frame.

2.2 Need of Empirical Studies

Empirical studies generally consider the data analysis methods and statistical techniques for exploring the relationships between various factors of the problem domain by using the data collected through experiments. Empirical studies are important because it allows the exploration of relationships among independent and dependent factors. It helps in providing proof to the theoretical concepts through experimental data. It helps in accurate evaluation of the proposed pathfinding algorithm or the data structure. The empirical study enables us to choose from various techniques and data analysis tools for generating meaningful results. Empirical studies are essential in the area of pathfinding as it allows to evaluate and assess the new ideas, concepts, algorithms, tools, strategies, heuristic and data structure in scientific and proved manner. It also facilitates improving, controlling and managing the existing methods, pathfinding techniques, and strategies by using evidence from the empirical analysis. The information collected from empirical studies helps in decision making and understanding of the concepts of the pathfinding techniques. It will help in building quality benchmarks for future experimentation across pathfinding community and more reliable results can be obtained by standard comparison of the proposed method with the previous methods. The empirical studies are also beneficial for game developers in selecting the appropriate search algorithm for their game while giving them enough space to focus on other aspects of game development. The empirical study enables the gathering of evidence to favor the claims of a technique or proposed method. Therefore, it is necessary to conduct

empirical research for building a knowledge base so that high-end algorithms can be developed and utilized.

2.3 Problems with Current Research Practices

Every year many research papers are published in every field. In computer science, the work in research papers is supported by experimental data and analysis of that data. Then the results are concluded and presented in the research papers. Sometimes the results given in research papers do not seem reliable according to the provided information. Therefore, after surveying and critically analyzing more than 150 papers, this thesis highlights the problems with current practices. The first problem is the lack of causal explanation of the behavior of the proposed algorithm or data structure. Many papers do not try to explain the reasons behind the performance of their algorithm. The experimental results show that the algorithm works but often doesn't know how to determine whether in what conditions it works well or poorly. Many papers claim that their algorithm works better than the other but usually can't attribute the difference in performance to the algorithm or differentiate the influence of algorithm from the influence of the experimental setup. As papers lack causal explanations thus, their results are quite often misleading means they claim things which are not true. There are high chances that something goes wrong with experiments as compared to theory because theory is based on logical proofs, but experiments are like cooking recipes, slight difference in ingredients could lead to entirely different result [12]. Therefore, a minor mistake while conducting experiment can generate misleading results. Experiments are easier to mess up than the theoretical proofs.

Advancement in technology and science makes the problems more complex. Therefore, it is not possible to find theory-based solutions supported by complex theorems and some problems are empirical in nature. So, it is better to conduct

empirical study to find solutions to the problems with the help of experiments. As mentioned earlier that experiments can go wrong and generate misleading results which turns into incorrect interpretation of data. Another problem is the bias results, means the experimental setup is in favor of the proposed method. In pathfinding, maps are often generated randomly and does not provide real world scenario and the results can be biased with the generation of simple maps rather than complex ones. There is no standard setup or guidelines exist for pathfinding therefore, the comparison with others work is not at the same scale. Everyone uses their own set of maps for testing algorithm's performance, so it is hard to do comparative analysis of their work. These are some basic issues with the current practices in pathfinding research.

2.4 Critical Review of Pathfinding Research Papers

Research papers are an important part of current developments in pathfinding area, as it provides insight into the work done in past and future expectations. It is necessary to critically review the research papers published in past in order to make future research more valuable and reliable by learning from their mistakes and weaknesses. In this thesis nearly 150 research papers are critically reviewed during literature study and throughout our thesis work to find out the pitfalls and problem areas while conducting research. Critical review of some papers is presented in this section. Kai Li et al. [11] proposed a boundary iterative-deepening depth-first search (BIDDFS) algorithm which repeats its search from the saved boundary location, minimizing the search redundancy in most of the iterative search algorithms (repeats its search from starting point each time). The experimental results only show the time taken and threshold, in which Dijkstra beats the time of BIDDFS, but it does not provide clear evidence like the memory usage. Therefore, the paper fails to provide causal explanation of their algorithms performance which might result in mislead

interpretation of data. Also, the paper gave no evidence that their algorithm had been tried on more than one set of maps (randomly generated square maps), no real-world maps were used to test the algorithm's performance. Another problem was that BIDDFS had the same threshold as of IDDFS in all the cases and takes more runtime for lower obstacle density. Threshold was described as memory efficiency but the paper did not provide any memory comparisons, which was the claim, that proposed algorithm consumes less memory.

The research paper written by Yngvi et al. [8] proposed two new effective heuristics for A*. The first one is, the dead – end heuristic, that reduces the search area from the map which is irrelevant to the current path query and thus claimed to be more effective than general octile heuristic. The second heuristic, called gateways heuristic, used the decomposed map from the previous heuristic, then consider the boundaries of the omitted areas as gateways to pre-process the path from one gateway to all other gateways and thus, claimed to better estimate the path cost. Now, the way this paper was presented has three main areas where it lacks empirically or did not provide enough information to the readers. Firstly, it uses one demo map and nine game maps, but did not provide any information regarding the range of map size, obstacle density and distribution of obstacles. Secondly, the author did admit that the proposed heuristics use extra memory for pre – calculations but did not give any range or number for the memory usage. Thirdly, the author claims that heuristics take less time for the final pathfinding but did not provide any data for the time taken by these heuristics for pre-processing of the map decomposition and distance between gateways. Whereas, in octile heuristic neither pre-calculation nor decomposing the map is required. So, from reader's perspective this paper did not answer all the questions arising in reader's mind.

In our thesis, we tried to create case studies surrounding these problems and provide some solutions. As it is not possible to recreate same experiments because every paper uses discrete setups, we generated general problem cases to cover these problems.

2.5 Summarizing the critically reviewed papers

The pathfinding research papers reviewed for our thesis were analyzed based on the information provided by the author in the paper. As we do not have access to the researcher's data, we assume that the author provides the information to the best of their knowledge and supporting data. Therefore, based on the information in the research paper, we find that most of the papers have poor experimental setups, inappropriate collection of data and unreliable data analysis techniques or representations. In most of the experimental setups only one type of map is used and most commonly used maps are either random maps or game maps. These two types of maps are used by most of the researchers because random maps are easy to generate and also obstacles are placed randomly, which they assume will covers the typical problem area and game maps are used by some researchers for different reasons. But in reality, random maps do not provide enough challenge to test the algorithm efficiency. Some of the papers also use only one size maps, like paper written by David [14] used only maps of size 32 x 32 and another paper used only maps of size 300 x 300 [15], therefore making it difficult to compare the work of these papers and also this kind of experimental did not provide enough evidence to support the performance of their algorithm. Both of them randomly generated their maps and obstacles are also distributed randomly. David [14] also used only 20% obstacle density which again is not a good example of testing algorithm's efficiency.

Although some papers are written very well and have some good experiment designs, but these researchers are very experienced researchers in this field and conducting experiments for many years thus have learned from their past experience. Most of these papers are written by Nathan R. Sturtevant, Robert C Holte, and Jonathan, working for the past decade in this field. For example, Paper written by Nathan et al. titled, "Real-time Heuristic search for pathfinding in video games" reviews three modern algorithms empirically and provide detailed analysis of the performance of these algorithms. This paper highlights both the positives and negatives of these algorithms, based on the empirical evidence and gives meaningful insights and information about the working of these algorithms. Although this paper used only game maps for their experiments but it was written for games so it justifies its purpose. Other than that it was very well written and consider the statistical methods to analyze the final results, as standard deviation is calculated along with simple mean and median to address the issue of variance[16].

And lastly, some of the papers we reviewed are theoretical thus we cannot not comment on those papers because there is no empirical evaluation of their research in the paper. For example, paper written by Peter et al. This paper highlights the underlying formal mathematical theory of incorporating heuristic information into graph searching. This paper proposed A* algorithm which uses the heuristic information to make an informed decision about expanding next node. The authors provide theoretical proof that with the given heuristic information their proposed algorithm is bound to find the optimal path from the start node to the goal node. [17].

CHAPTER 3

Empirical Research in Pathfinding

3.1 Motivation

Pathfinding problem is popular among AI society and lot of work was published to address this problem. While reading literature, we realized that some of the papers lack experimental evidence supporting their claim. Then papers were reviewed critically for more information and it was clear that some of the results were misleading and lack explanation. The problem was the absence of clear set of guidelines to conduct research in pathfinding. When some more literature study was conducted, we found that experiments are very volatile and can be easily messed up, which could lead to wrong interpretations and conclusions, we decided to find the solution and recommend some methods to avoid the pitfalls while conducting empirical study.

3.2 Empirical Research and its basic components

Empirical research is the combination of exploratory techniques (visualization, summarization, exploration and modelling) and confirmatory procedures (testing hypotheses and predictions) [12]. Empirical study is important for pathfinding research as it allows researchers to evaluate and assess the new algorithms, techniques, methods and concepts in scientific and proved manner. It also facilitates the improvement and management of the existing methods by using evidence collected from empirical analysis. According to Cohen [12], there are six basic components of empirical research which are Agent (proposed method), task

(performance evaluation), environment (maps or test subject), protocol (experimental setup), data collection and analysis. In pathfinding, agent is the proposed algorithm or data structure, task is to evaluate its performance, environment is maps or graphs, protocol is experimental rules and setup, data collection is record of experimental results and then analysis of those results. The first three represents the theories of behavior, the last three are part of empirical study. According to Malhotra [13], there are four basic elements of empirical research which are mainly purpose, participants, process and final product. Purpose refers to the motivation of the research, means building the research question and the reason for conducting research. It basically means asking a question, "Why are we conducting this research, why is it so important?". The next element is the participants, around which a research work must be done, the matter of the research. It is very important to handle the participants with adequate manner, especially in computer science all the ethical issues should be managed properly. The process gives the details of the steps to be taken in order to conduct a research in rightful manner. All the research details like planning, literature, techniques, programs and methodologies to be used and the sequence these must be conducted constitutes the process of research.

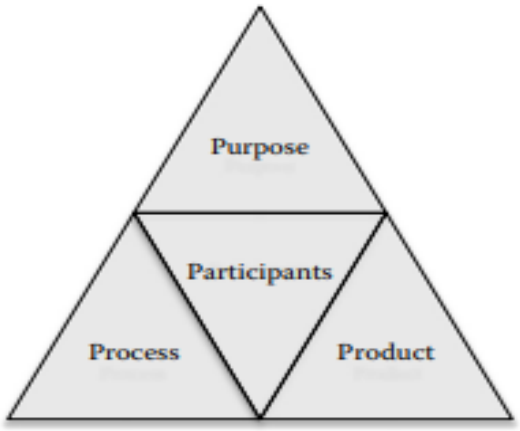


Figure 5: Elements of Empirical Research[13]

Finally, the last element is the product, which in computer science research is the conclusion and results being calculated from the data collected during the process. These results then help the researcher to answer their research question asked in the beginning of the research.

3.3 Guidelines for Empirical Research

In empirical Research it is important to understand the issues and pitfalls related to it. So, here we underline some general guidelines to conduct empirical research.

3.3.1 Objective

The very first stage of the empirical research is defining the goals of the study. The objective must be clearly defined and explained in detail to provide meaningful information. It must dispense the purpose of the study like researcher's final goal and sub goals, the areas to be studied and its impact, the system to be used and, the motive of the study. It must answer the following questions:

- What is the aim of the research?
- What are the areas of focus?
- What type of research is being conducted?
- What techniques or programs are being studied?

These questions can help to precisely define the objective of empirical study. The research question should be exploratory in nature rather than comparative, means the question should look for possible explanations of the performance not on comparing which performs better. It is equally important to make a clear hypothesis question as conducting the experiments. If the hypothesis is not clearly defined, then experimental results will not provide useful data. For example, finding which algorithm finds better path is not a valid research question but why an algorithm finds

better path than other is a valid research study. Finding out the reasons for the algorithm's performance will help us in identifying the areas which affects the performance of the algorithm so that we can work on them in more detail or manipulate those parameters to yield better performance. Taking the first step correct makes next steps clearer and easier. So, defining the research question is our first step and it must be precisely designed. Literature study plays an important role in defining our research problem and based on that background literature research one can formulate the research question for their empirical study.

3.3.2 Experimental Setup

The next stage in empirical research is conducting experiments to test the algorithms and generate data for further analysis to draw solid conclusions. The experiment should be designed according to the objective of study. It must cover all the prospects of research question, like formulating the appropriate hypothesis to be tested, figuring out the variables to be used, type of data to be collected, analysis techniques to be used and last how to represent that data.

The initial stage of experimental setup requires formulating the research question. This means that it should address the concepts and relations to explore in the research. The research question will be like: What attributes of the map will affect the performance of the algorithm? Or What is the impact of the data structure on the working of the pathfinding algorithm? The research question should address the area to be studied for the problem, which in turn help us identifying the independent and dependent variables. Independent variables are defined as the variables that can be controlled or manipulated during the experiment while dependent variables are the output variables that depend on the independent variable. Any change in the independent variable directly affect the dependent variable. If the problem is clearly stated then it makes it bit easy to define which variables to be considered dependent and which ones to be independent. For example, in pathfinding, map attributes like –

size, obstacles density, distribution of obstacles are independent variables and the time taken for searching the map, number of operations, nodes expanded are dependent variables. The path length is a special attribute because the path length somehow depends on the map size but within the map it is considered independent variable as we can change the path length by changing the start and goal positions. In this stage researcher should also think about the analysis methods to be used to estimate the amount of data to be generated and collected. By going through the analysis before executing the experiment. The next important thing is to state the environment of the study, language used for programming, maps used, algorithms used, data structure and so on.

3.3.3 Data Collection

The researcher should make decision on the sources of collecting the data for analysis, means that the data is generated by conducting the experiments only or using the available benchmark datasets or from the literature. If using both benchmarks sets and experimental data, then also state how the data is collected like if the researcher used the same experimental setup as of benchmark or the literature setup with which the data will be compared. If the researcher used different experimental setup like different programming language, different IDE or platform, then how the comparison with previous data will be validated. Is it a direct comparison, which we do not recommend, or indirect comparison like first by collecting the similar data set as of the benchmark set and analyze it, if it concludes the same result as of the benchmark or literature result then further conclusions can be drawn based on these observations? Researcher should avoid collecting too much or too little data, because if large or less data is collected then we can miss important information. Try to collect only relevant data and informational data. Initially a large data is collected and then it should be normalized or reduced by applying some formulas or various statistical techniques like averaging, standard deviation and so on, which makes analysis of the data more convenient.

3.3.4 Data Analysis

Data analysis stage plays a crucial role in answering our research question. It involves the understanding of the data by reducing it suitably and which can be read easily. Then this reduced data can be used for further analysis. But data must be reduced carefully without losing its original information. One way of reducing the data is dividing the dependent variables into different categories and then conduct the comparison among those categories. The second way is using some sort of statistical measures such as mean, median, or standard deviation. While using these averaging statistical techniques, the outliers must be assessed with precaution because sometimes outliers provide useful insight into the performance of the algorithm. After reducing the data, statistical techniques can be used like linear regression, logistic regression, and so on for producing analyzing charts. Analysis method should be selected based on the research question means it should provide meaningful answer to the question asked at the beginning of the research.

3.3.5 Representing results

Finally, after analyzing the results reasoning must be provided for the explanation of the answer using the data. Results must be represented in appropriate format and from readers perspective. The report in which the results are represented must clearly document the background, motivation, experimental design, analysis and results. To represent the result bar graphs, line charts, pie charts or other methods of representation can be used, which reader should easily understand. Although the results should be represented in simple form, for readers prospective, but it should also retain the important information. The report must answer all the readers question, it must provide significant details about every aspect of the research. The results should be represented using an appropriate chart. It is not a good idea to put all the results in a table because tables just show the numbers not the actual trend in

the data or some relationship between the parameters. So, it is recommended to use bar charts or some sort of other representation of data along with tables.

In our research we followed these above-mentioned guidelines. Firstly, based on literature study we find our research problem and formulate our research question which was, ‘How to empirically evaluate the performance of pathfinding algorithm?’. After that we did some more literature study and find out the core area that we want to study which was exploiting maps and their features for algorithm performance evaluation. We also want to address the issues in evaluating the pathfinding algorithms. So, we design our experimental setup accordingly. Before designing the experiments, we figure out independent and dependent factors, like map size, density and distribution of obstacles as independent factors and time, number of nodes, path length as our dependent variables, so that our experiments are more precise. Also, we roughly layout the amount and type of data to be collected and data analysis techniques before working on experimental setups. Thus, it saves a lot of time and effort as we did not generate and collected irrelevant data. These general guidelines help a lot in empirical research where a lot of experiments are conducted to verify the results.

CHAPTER 4

Case Studies and Results

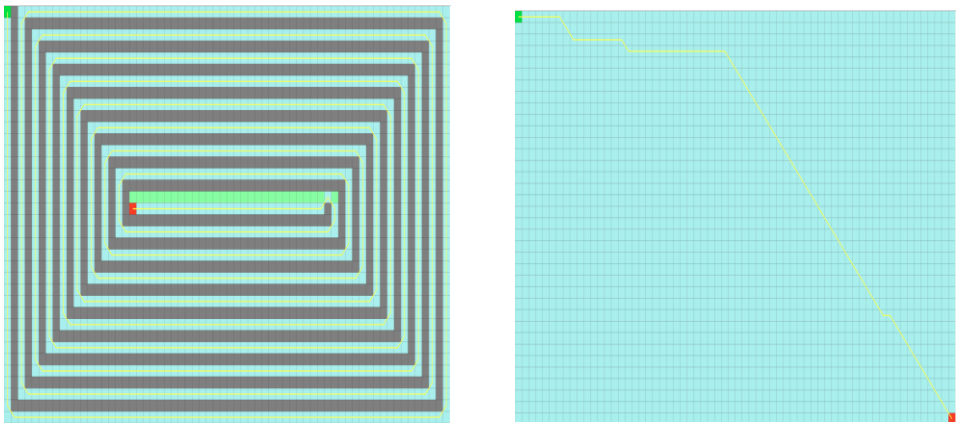
4.1 Pitfalls in empirical studies

Although while reading a research paper it seems that researcher conducted the study with clarity from the beginning of the research. However, this is not the case, conducting a research is a long and raw process which started with a vague idea of research problem, making mistakes, correcting them and learning from those mistakes, and finally redefining the research problem so that readers can understand easily. When conducting research one can have some problems which in the end can affect the results or answer to the research question. One can think that scientist or researchers are perfect in doing experiments and know everything about the experiments but in reality, even they made mistakes. The following example from an article published in 1991 will explain that how experiments conducted by scientists or anyone can go wrong sometimes. In 1991 New Scientist published an article about the search for an AIDS vaccine. The article says that initially scientists from Britain's Medical Research Council developed the vaccine using two components, human T cells and SIV virus. They first infected the cells with the SIV virus and then inactivate the virus and prepared vaccine from it. They gave this vaccine to four macaques and then gave them a live virus. It turns out that three out of four were protected against the virus. Later they conducted another research in which they gave uninfected human T cells of the same type to other four monkeys and then gave live virus, and the results show that two out of four were protected. So, when this study was published many other researchers said that the later scenario should have been considered from the very beginning of this study. However, the scientist from MRC defended that this possibility did not seem obvious at the beginning [14]. One can

think that they should consider all the possible variables in the beginning before making their claim that the vaccine created from T cells and virus will work on AIDS, but it is not possible to control all the variable possibilities directly. One can handle very few variables at a time, so there is always the possibility of error and encountering the pitfall while doing experiments [15]. The common pitfalls in empirical study are the ceiling, floor, regression, order effects, which are commonly known as Spurious effects, control conditions, sampling bias, collecting and analyzing results. Spurious effects are defined as the mathematical relationship between two or more variables in which they seem to be associated directly but in reality, it is either by chance or due to some other factor. Basically, the spurious effects make some results seem to be more effective when they are not or vice versa. These pitfalls are discussed in the following section:

4.1.1 Spurious effects:

Floor effects are described as the worst-case scenario in which the algorithm's performance is as bad as possible whereas ceiling effects arise when algorithm performs as best as possible. In pathfinding the ceiling effect occurs when an algorithm expands near or same number of nodes as the path length.



(a) (b)

Figure 6: Ceiling and Floor Effects in Pathfinding

In fig 9. (a) the ceiling effect occurs because the algorithm expands nearly same number of nodes as path length because those are the only traversable nodes in the map. One can claim with this map that their algorithm works best as it gives the maximum path length with almost same number of nodes expanded in high obstacle density map. But in this scenario even the simplest pathfinding algorithm will perform efficiently. We should not avoid these maps; we can use them because this kind of maps will give us the ceiling value by giving the minimum number of nodes expanded for the maximum path length. Fig 9. (b) shows the floor effect in which the algorithm traverses the whole map means expanding all the possible nodes for the given path length. Therefore, calculating these values help us analyzing our results more accurately. How to use these values is explained in the later sections of this chapter.

Next is the regression effect which is defined as getting extreme value on the first run and lower values or near average values on later runs. For example, when we are comparing the two heuristics Octile and Euclidean for A* algorithm, our hypothesis is that Euclidean takes less time than octile in finding the path. In order to do that we run our algorithm first with octile as heuristic for 10 times and record the values as shown in table 1 (a). Then we set the criteria that we select those instances in which the time taken is more than 1.5, because we think that these are more complex as compared to others, and run our algorithm with Euclidean as heuristic function on those instances again and record the values as shown in table 1 (b). Now according to these observations, it is concluded that our heuristic is true, thus proves that Euclidean takes less time than octile. But in general, Octile takes less time than Euclidean. What happens here is that the values we got on those three instances are higher because of either noise or extraneous factors and when we run the same algorithm with octile on those instances again we get lower values as shown in table 2.

A* Octile	
Length	Time
46	2.5
60	1.1
49	0.8
66	0.7
52	0.5
70	0.4
46	0.8
76	1.3
38	1.9
65	2.2

(a)

A* Euclidean	
Length	Time
46	0.9
38	0.7
65	1.4

(b)

Table 1: Example of regression effect using Octile and Euclidean heuristic.

A* Octile					
Length	Time T1	T2	T3	T4	T5
46	2.5	1	0.7	0.8	0.6
38	1.9	0.8	0.6	0.7	0.5
65	2.2	0.7	0.9	0.8	0.8

Table 2: Octile heuristic with 5 run times for each instance

4.1.2 Control conditions:

As mentioned earlier, MRC researcher problem is a good example of control condition. This is another pitfall in empirical studies as it is not possible to control all the variables affecting the dependent variable. Firstly, there are basically three types of variables which affect the dependent variable, one is obviously the independent variable which we manipulate during the experiment, other is the extraneous variable which we can control directly and the last one is the noise variable which we cannot control directly. Extraneous variables are the variables that influence the results of dependent variable along with the independent variable. Extraneous variables can be controlled directly or indirectly, by considering them as noise variable, through random sampling. For example, in pathfinding when we execute the algorithm to find the path, the processor, at that time simultaneously is running other backend applications which makes sometimes our runtimes longer than actual time required to process it. Therefore, we can control it directly, first by recording the data with backend applications running, then stop these applications and run our program again and record the data. Now we have both the readings so we can compare these values to see if it actually is influencing our results. Other possibility is to consider it as noise variable, in this case we will run our algorithm with same start and goal node several times and then randomly select few values to get our average running times.

4.1.3 Sampling Bias:

Sampling bias means the data collected for analysis represents certain group of instances not all the possible ones, which results in analyzing outputs in favor of the problem under study. This issue arises either because of control conditions or by selecting some specific data based on some criteria like setting a threshold value. If we cannot consider all the variables or possibilities while conducting or designing experiments then we will make the probability of selecting some specific instances

zero which is sampling bias. Even if we designed our experiment sound and consider all the scenarios, we could generate a biased sample by selecting data from certain threshold value or based on some specific criteria which eliminates a certain type of data to be included in analysis. A general example to understand this is given as a survey of coffee shop in downtown to measure the number of people visiting coffee shop in a day will be biased as majority of the people will be working in nearby offices which does not include the people outside the downtown area. The coffee shop will be busier than the similar coffee shops in other areas of the city. The example in pathfinding is, if we select data from specific kind of maps such as larger size maps randomly generated then the results will be biased because it does not include other types of maps such as maze maps, game maps, city map and also the maps smaller in size.

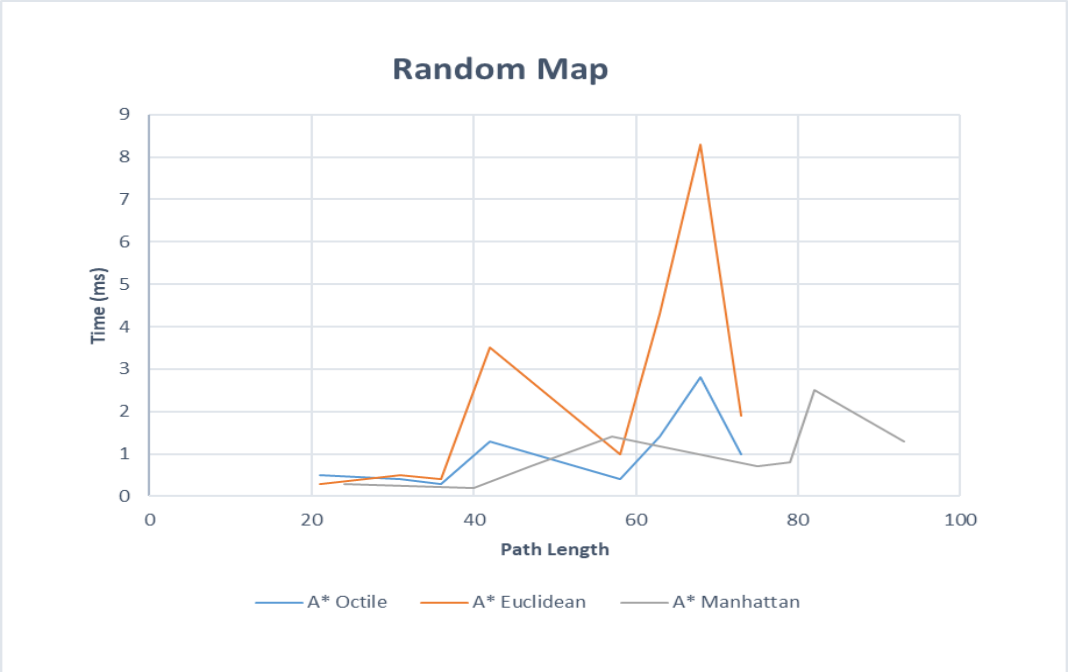


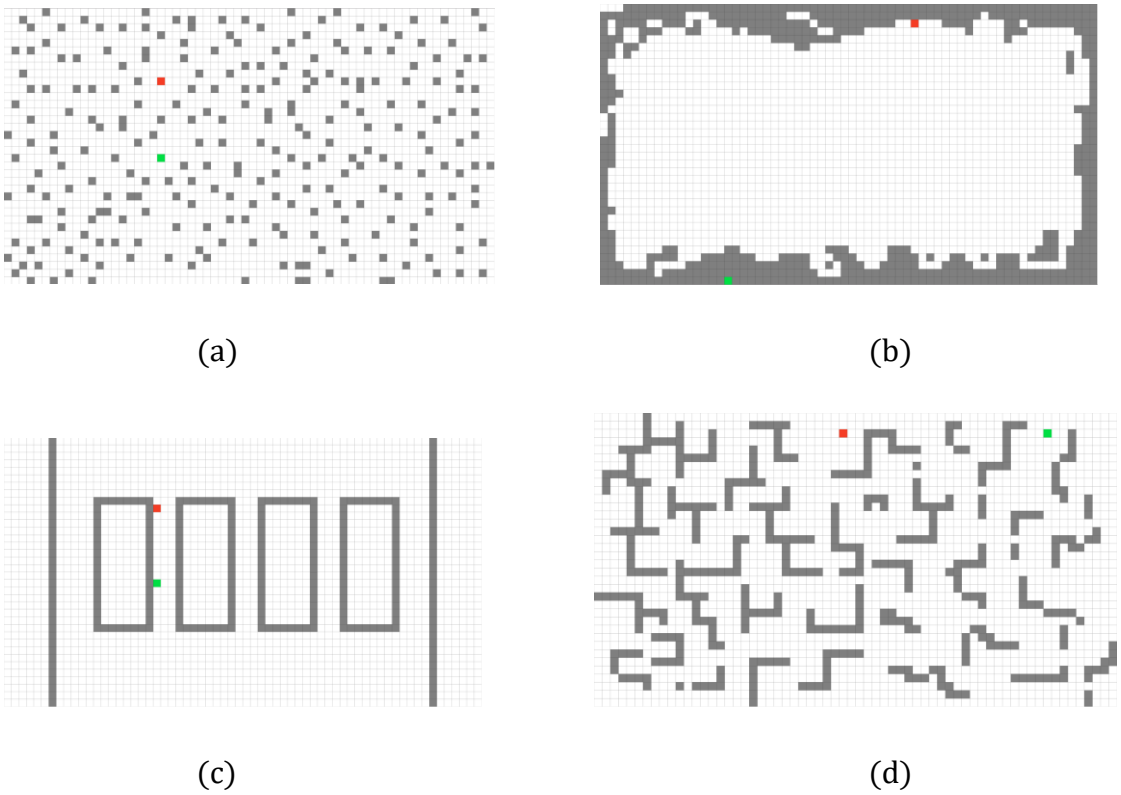
Figure 7: Time and path length results on Random Map size 120 X 120 with obstacle density 15%

In fig 10 and 11, we can see that the results for Euclidean heuristic are more extreme in Random Map as compared to the results obtained in terrain map. Now, collecting or analyzing results from only Random Maps will give us biased opinion on the performance of Euclidean Heuristic.

Figure 8: Time and path length results on terrain map size 120 X 120 with obstacle density 15%

4.2 Experimental Setup:

Our thesis is about setting guidelines to conduct empirical research in pathfinding based on Maps. We used only 2D grid maps because most of the researchers use these maps. We classified maps into four categories: Random Maps, maps generated randomly, Game/Maze Maps, maps from games like Dragon Age 2, Baldurs Gate 2, maps of various mazes, Room/ Floor maps, maps of rooms, building floor maps and Terrain Maps, maps of real world like parks or city maps with obstacle density ranging from 0% to 50%. The reason for choosing different categories is to include the different distribution of obstacles on the maps which is an important factor affecting the dependent variables.

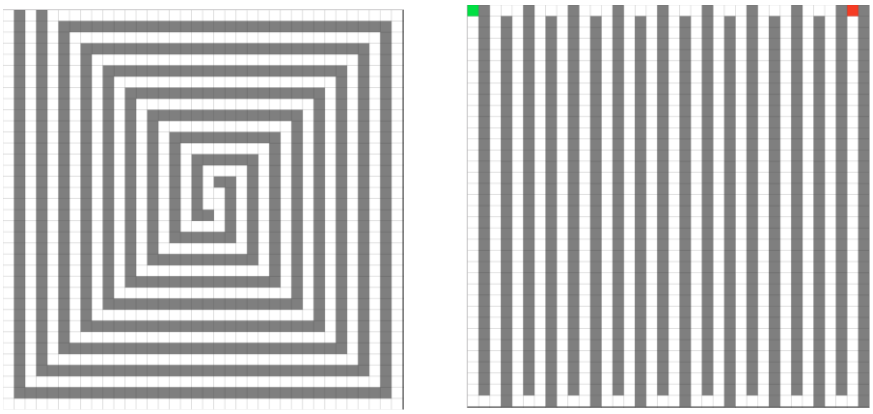


(a) (b)

(c) (d)

Figure 9: Maps of size 120 X 80 with obstacle density 15%. (a) Random Map, (b) Terrain Map, (c) Floor Plan Map, (d) Maze Map

When we increase the density to 50% and the obstacles are distributed in a certain way, we can get the maximum path length possible in the given map. In fig. 10 three different distributions with 50% obstacle density will give maximum path length. We cannot increase the density of the obstacles in these maps because if we do so then path will be blocked for some start and goal positions and if we decrease the density then the path length will decrease.



(a) (b)

Figure 10: Maps with 50% obstacle density with maximum path length.

The algorithms implemented for our thesis are Dijkstra and A* with three different heuristics: Manhattan, Octile, and Euclidean. The data is generated and collected based on the problem cases and pitfalls we want to study.

4.3 Solutions to Pitfalls

The pitfalls mentioned earlier will result in producing analysis reports not representing actual performance of the algorithm. Therefore, we will suggest some solutions and methods to handle these pitfalls through case studies of problems that follow the suggested solutions.

4.3.1 Case 1 – Four Spurious effects:

The four spurious effects discussed earlier, if not addressed, will conclude different results than actual. The map used to represent the ceiling effect and maps in fig. 13 are special maps because in these maps' obstacle density is high and also gives the

maximum path length with minimum number of nodes expanded. Thus, finding the ratio of number of nodes expanded and actual path length will give us the ceiling value (c). The ratio (R) will be given by:

$$R = \frac{Number\ of\ nodes\ expanded\ (n)}{Actual\ Path\ length\ (l)}$$

The ceiling value in ideal situation will be 1. In pathfinding the ceiling value is known or we can say constant, but it is hard to find one constant or single floor value because that depends on number of factors like map size, obstacle density and distribution of obstacles. But, in general floor value will be exponentially high than actual path length. In worst case, an algorithm will explore every open node in order to reach the goal node like Dijkstra algorithm explores all the nodes before reaching the goal node as shown in fig. 6(b). If we have map of size 10 x 10 with no obstacle, the start node is at the upper left corner and the goal node is at lower right corner then the path length will be the diagonal between these two nodes which is 10 and the algorithm will explore all the nodes in the map which is 100. Thus, from this situation we can generalize the floor value to be n (map size).

The solution to the regression effect is instead of choosing values over certain threshold (like 1.5 as mentioned in regression problem), we should sort the data first and then make some criteria of randomly choosing every third or fifth value from the data to test our next algorithm, in case if we do not want to run the new algorithm on all the instances again. This will ensure that the new algorithm will get evenly distributed instances not just above threshold instances. Although, we recommend to run the new algorithm on all the instances used in the previous algorithm.

A* Octile											Average
Length	38	46	46	49	52	60	65	66	70	76	
Time	1.9	2.5	0.8	0.8	0.5	1.1	2.2	0.7	0.4	1.3	1.22

Table 3: A* octile run times sorted according path length and their average

A* Euclidean	
Length	Time
46	0.8
60	1.9
70	1.5
Average	1.4

Table 4: A* Euclidean run times when every third instance from the octile runs is selected.

As we can see in tables 3 and 4 that when we sort the data of A* with octile heuristic and select every third instance and run the A* with Euclidean heuristic, the average run time of Euclidean is higher than octile.

4.3.2 Case 2 – Control conditions:

In order to solve the problem of control condition we have to follow the general guidelines for experimental setup because when we know our research question and parameters used to collect data, then we can easily identify the independent and extraneous variables. Thus, we can manipulate them or control them directly to get the true influence of independent variables on the variables to be studied. There are

two ways to solve the problem of control condition mentioned in earlier example, one is stopping the backend applications and making the processor free from any other activities and run our program to collect the data. The other possible solution is to consider this as noise factor and run our algorithm several times with same start and goal position, then take the average. For example, we run the octile heuristic five times for each start and goal position and then take the average of these five runs. When we replace the run time with these new average run times, we get lower value of overall average run time reduced from 1.22 to 0.86 as shown in table 5.

A* Octile											Average
Length	38	46	46	49	52	60	65	66	70	76	
Time	0.9	1.1	0.8	0.8	0.5	1.1	1	0.7	0.4	1.3	0.86

Table 5: A* octile with new average run times for each instance.

Another factor to control in pathfinding is the distribution of obstacles in the map. So, for this we can use the four types of maps given in fig. 12. As these maps will cover most of the distributions we can encounter in real world or games or in random maps.

4.3.3 Case 3 – Sampling Bias:

To reduce the effect of sampling bias we should generate different kinds of maps with different range of size, obstacle density and path length. After that combine the data and represent that information on chart to get the performance measure of the algorithm.

The chart in fig. 14 combines the data from all the maps of different types and it shows the running times along with path length for A* with three different heuristics. Although this chart does not provide any clear picture about which heuristic takes less time or clear lines for the time taken by each. There is another way of representing the same data as shown in fig. 15.

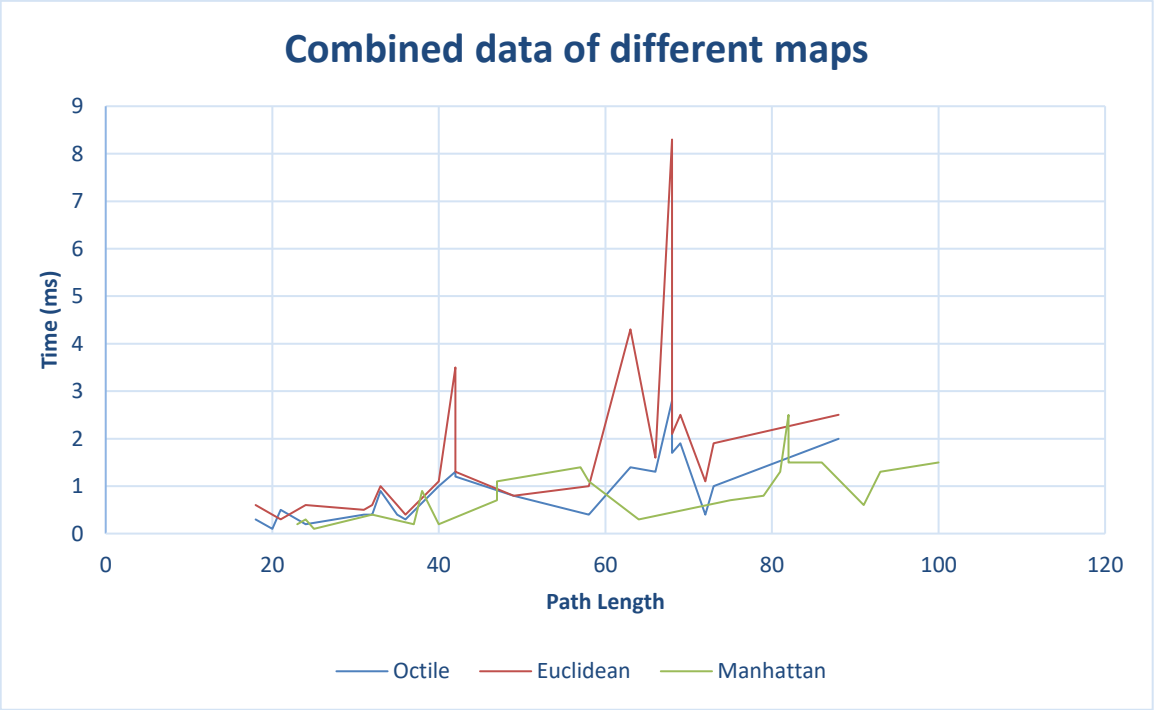


Figure 11: Chart showing combined data for A* from different maps of size 120 X 120 and obstacle density is 15%.

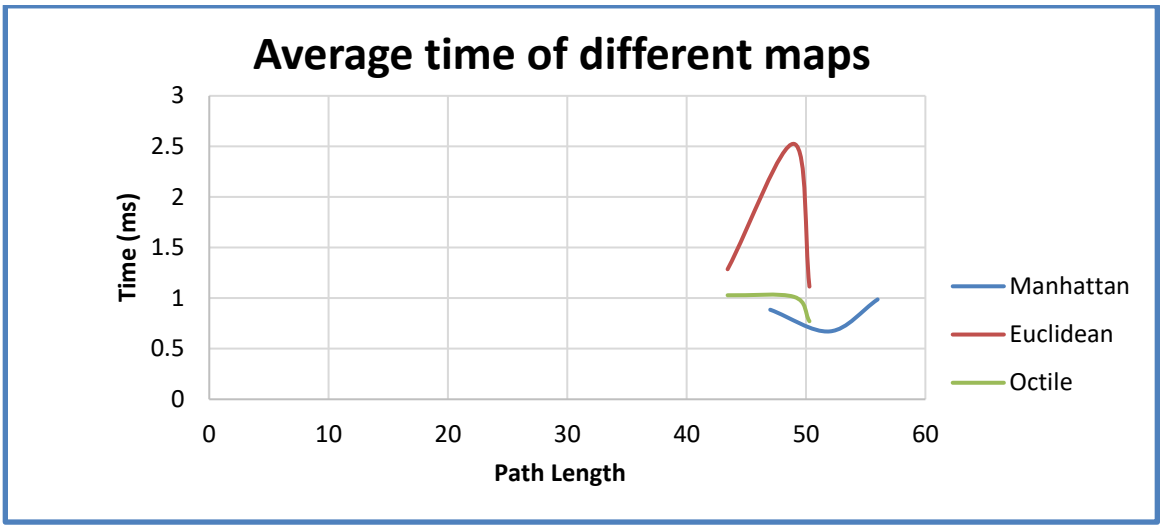


Figure 12: Average time and path length of four different types of maps, size: 120 X 120, obstacle density is 15%

In fig. 15 we calculated the average time and path length of each type of map for the three heuristics and then projected it on chart. Now, from this chart we can easily conclude that Euclidean takes more time on an average than other heuristics and Manhattan takes less time but it gives longer path lengths as it can only move in vertical and horizontal directions not diagonally as other two heuristics.

We can also use bar charts or other representations based on our data set and parameters, but we should make sure that it represents valid results not the biased or misrepresented results.

4.3.4 Case 4 – Example problem from research papers:

For our thesis we critically reviewed the literature published in the field of pathfinding. Based on that we are representing one example problem that was mentioned in chapter two section 2.4, the research paper written by Yngvi et al [8].

We used their data given in their paper and tried different representation. The author represented the averaged data in the form of table as given in table 6. We projected their data on scatter plot chart displaying the number of nodes expanded and the time taken to calculate the final path in their experimental maps. As the authors did not provide the time taken for the pre – processing of their map decomposition and gateway calculations, we find that the difference in time and nodes expanded does not seem to be significant as shown in fig. 13.

Now, by just reading the paper it is hard to conclude any result, one need more information to convincingly deduce or get reliable results. Also, it is hard to reproduce results of many papers as they use different experimental setups

	Demo map	**Octile**	**Dead - End**	**Gateway**
All	path cost	7430	7430	7430
	estimate	3940	3940	7241
	nodes	955	579	220
	time (ms)	18.6	14.7	13.2
top 10%	path cost	14373	14373	14373
	estimate	6605	6605	14179
	nodes	2397	1352	487
	time (ms)	42.9	30.4	28
	Game maps	**Octile**	**Dead - End**	**Gateway**
All	path cost	10339	10339	10339
	estimate	7788	7788	9884
	nodes	1231	1120	723
	time (ms)	27.3	24.6	22.6
top 10%	path cost	20468	20468	20468
	estimate	13290	13290	19731
	nodes	3701	3370	2313
	time (ms)	69.2	60.7	54.5

	Large map	Octile	Dead - End	Gateway
top 10%	path cost	30463	30463	30463
	estimate	17201	17201	30002
	nodes	5961	4536	2361
	time (ms)	110.1	84	71.3

Table 6: Data as given in Yngvi et al. research paper for the three heuristics.[8]

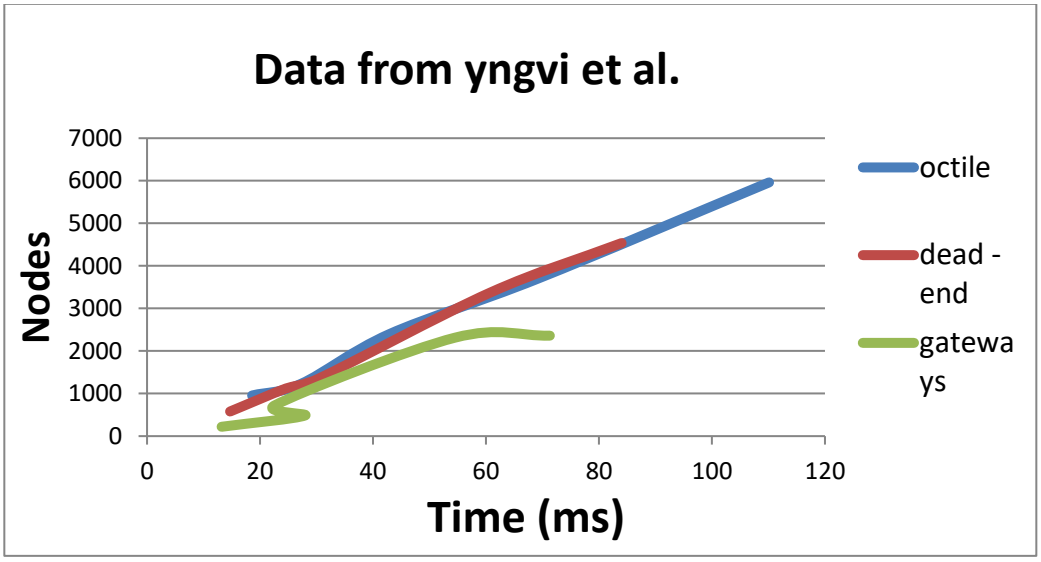


Figure 13: Different Representation of yngvi et al. data in the scatter plot chart.

CHAPTER 5

Conclusion

In our thesis we proposed some guidelines and solutions to pitfalls that we discovered while reading research papers. The sample problems for each case study based on pitfalls present solutions with relevant experimental data. Our research shows that the pitfalls in pathfinding can create misleading results. The main focus of our research is only 2D grid maps, not the algorithms. Different features of maps are manipulated to generate different maps, like the size of maps, the density of obstacles in the map, distributing obstacles in different possible ways and then generate the data using random start and goal positions. All these features are the independent factors which we can exploit and manipulate, then we record their effects on dependent variable like run time, number of nodes expanded, number of operations, path length. We first provide some general guidelines, according to which we should first try to formulate the research question or narrow down our problem domain so that we can design the experiment more precisely which will result in providing meaningful data. We strongly recommend going through data collection and analysis method before implementing the experimental setup because that will help us determining the independent and dependent factors and also help us with the amount of data to be collected for our research. This will save our time and resources which we can utilize somewhere else. Our research shows that even if we design good experiment, one can still conclude unreliable results because of the pitfalls mentioned in chapter four.

Our research shows that encountering these pitfalls give us false results and can be misleading for future research work. Although it is not a serious issue in games or other fields related to computer science but if we implement these algorithms, based

on unreliable results, in real world applications like GPS and other direction providing services it could be life threatening.

CHAPTER 6

Future Work

Our research only explores the 2D grid maps and their associated features like path length, map size, obstacle density, distribution of obstacles. In our research, we summarized and critically analyzed the research work done by others in this field. Based on the review we outline some guidelines and case studies to overcome the issues found in their research work. But we still have to explore the 3D maps and other representations of the maps like navigation mesh, waypoints to set guidelines for conducting experiments using these representations. Also, we still have to explore various aspects of algorithms, underlying heuristics, various data structures and other components of pathfinding. After this extensive research, based on all these different aspects of pathfinding we then can generate benchmark problems and data sets, which will make future research work more comparable and reliable. There is still a lot to do in the field of pathfinding. In the future, more extensive research and empirical evaluation of pathfinding algorithm and its environment should be done to create the database for making the work more standardized and accessible.

REFERENCES

1. Algfoor, Z. A., Sunar, M.S., & Kolivand, H. "A comprehensive study on pathfinding techniques for robotics and video games." *International journal of Computer Games and Technology* (2015): 7.

2. Delling D., Sanders. "Engineering Route planning algorithms." Delling D., Sanders. *Algorithmics of large and complex networks*. Berlin, Heidelberg: Springer, 2009. 117-139.

3. Vinther, A.S.H., & Afshani, P. "Pathfinding in two dimesional worlds." Thesis. 2015.

4. Yap, P. "Grid-based Pathfinding." *Conference of the Canadian Society for Computational Studies of Intelligence*. Berlin, Heidelberg: Springer, 2002. 44-55.

5. Bjornsson, Y., Enzenberger, M., Holte, R. "Comparison of different grid abstractions for pathfinding on maps." *IJCAI*. 2003. 1511-1512.

6. Van Toll, W., Cook, A. F., & Geraerts, R. "Navigation meshes for realistic multi-layered environments." *International conference on Intelligent Robots and Systems*. 2011. 3526-3532.

7. Gutenschwager, K., Volker, S., Radtke, A. "The shortest path: Comparison of different approaches and implementations for the automatic routing of vehicles ." *Proceedings of the winter simulation conference*. 2012. 291.

8. Yngvi., Halldorsson, K. "Improved heuristics for optimal Pathfinding on Game maps." *AIIDE*. 2006. 6,9-14.

9. Pohl, I. "Heuristic search viewed as pathfinding in a graph ." *Artificial Intelligence*. 1970. 193-204.

10. Patel, Amit. *Amit's thought on pathfinding,*. n.d. 2019.

11. Kai Li LIm, Kah Phooi Seng, Lee Seng Yeong. "Uninformed pathfinding: A new approach." *Expert systems with applications*. 2015. 2722-2730.

12. Cohen, Paul. *Empirical Methods for Artificial Intelligence*. Cambridge, MA, USa: MIT Press, 1995.

13. Malhotra, Ruchika. *Empirical Research in Software Engineering: Concepts, Analysis and Applications*. Chapman & Hall/CRC, 2015.

14. Silver, David. "Cooperative Pathfinding." *Artificial Intelligence and Interactive Digital Entertainment Conference*. AIIDE, 2005. 117-122.

15. Cazenave, Tristan. "Optimizations of data structures, heuristics and algorithms for path-finding on maps." *IEEE symposium on Computational Intelligence and Games*. 2006. 27-33. (Bultiko)

16. Bultiko, Vadim, Yngvi, Nathan and Ramon Lawrence. "Real-time heuristic search for pathfinding in video games." Artificial Intelligence for Computer games. New York: Springer, 2011. 1-30.
17. Hart, Peter E., Nils J. Nilsson, and Bertram Raphael. "A formal basis for the heuristic determination of minimum cost paths." IEEE transactions on Systems Science and Cybernetics 4.2, 1968. 100-107.
18. “AI and Navigation,” Epic Games Inc., 2012. [Online]. Available: https://docs.unrealengine.com/udk/Three/AIAndNavigationHome.html. [Accessed May 2019]
19. “Int. J Computer games,” Games Technology., 2015. [Online] Available: https://www.semanticscholar.org/paper/A-Comprehensive-Study-on-Pathfinding-Techniques-for-Algfoor-Sunar/8de9c0429c106e46e7527074b4856d137f2f8b10/figure/1

APPENDICES

Appendix A

	List of pathfinding papers critically reviewed for thesis
1	A heuristic search algorithm with modifiable estimate
2	A combined tactical and strategic hierarchical learning framework in multi-agent games
3	A comparison between A* pathfinding and waypoint navigator algorithm
4	A comparison of high-level approaches for speeding up pathfinding
5	A Navigation meshes and real time dynamic planning for Virtual worlds
6	A formal basis for the heuristic determination of minimum cost paths
7	A game map complexity measure based on hamming distance
8	A heuristic for domain independent planning, and its use in an enforced hill-climbing algorithm
9	A hierarchical data structure for picture processing
10	A hierarchical data structure for representing the spatial decomposition of 3-D objects
11	A Comparative analysis of the algorithms for pathfinding in GPS systems
12	A hierarchical space indexing method
13	A note on two problems in connexion with graphs
14	A partial pathfinding using map abstraction and refinement
15	A polynomial-time algorithm for non-optimal multi-agent pathfinding.
16	An efficient memory bounded search method
17	A path planning algorithm for low-cost autonomous robot navigation in indoor environments
18	A∗-based pathfinding in modern computer games
19	Accelerated A* Trajectory Planning: Grid- based Path Planning comparison
20	Adaptive A*

21	AI game programming wisdom
22	An active wave computing based path finding approach for 3-D environment
23	An efficient and complete approach for cooperative path-finding.
24	An incremental algorithm for a generalization of the shortest path problem
25	An optimal routing strategy based on specifying shortest path
26	An overview of quadtrees, octrees, and related hierarchical data structures
27	Theta*: Any-angle path planning on grids
28	Anytime dynamic A*: An anytime, replanning algorithm
29	ARA*: Anytime A* with provable bounds on Sub-optimality
30	Artificial intelligence for games
31	Basic Point seeking: A family of dynamic pathfinding algorithms
32	Beamlet-like Data processing for Accelerated Path-planning using multiscale information of the environment
33	Fringe search: Beating A* at pathfinding on game maps
34	Benchmarks for grid-based pathfinding
35	Iterative expansion A*
36	Block A*: Database driven search with applications in Any-angle path planning
37	Reducing the search space for pathfinding in navigation meshes by using visibility tests
38	Case-based subgoaling in real time heuristic search for video game pathfinding
39	Comparing real-time and incremental heuristic search for real-time situated agents
40	Comparison of an Uninformed pathfinding: A new approach
41	Comparison of different grid abstractions for pathfinding on maps

42	A comparative study of navigation meshes
43	Complete algorithms for cooperative pathfinding problems
44	Comprehensive study on pathfinding techniques for robotics and video games
45	Contraction hierarchies: Faster and simpler hierarchical routing in road networks
46	Cooperative pathfinding
47	D* Lite
48	Database-driven real-time heuristic search in videogame pathfinding
49	Depth-first Iterative-Deepening: An optimal admissible tree search
50	DHPA* and SHPA*: Efficient Hierarchical Pathfinding in Dynamic and Static Game Worlds
51	Distance based goal ordering heuristics for Graph plan
52	Dynamic control in path-planning with real-time heuristics search
53	Dynamic path planning and movement control in pedestrian simulation
54	Efficient triangulation-based pathfinding
55	Efficient way finding in hierarchically regionalized spatial environments
56	Enhanced Iterative - Deepening search
57	Entropy and the complexity of the graphs
58	Euclidean heuristic optimization
59	Expressive AI: Games and artificial intelligence
60	Fast and Memory-Efficient Multi-Agent Pathfinding
61	Finding a pathfinder
62	Finding optimal solutions to cooperative pathfinding problems
63	Anytime heuristic search
64	Flight trajectory path planning
65	Fuzzy dijkstra algorithm for shortest path problem under uncertain environment
66	Generalized adaptive A*

67	Heuristic search viewed as pathfinding in a graph
68	Generic path planning for real-time applications
69	An improved pathfinding algorithm in RTS games
70	Geometric speed-up techniques for finding shortest paths in large sparse graphs
71	GPU accelerated pathfinding
72	Grid-based pathfinding
73	Heuristic collision-free path planning for an autonomous platform
74	Heuristic search in restricted memory
75	Hierarchical A*: Searching Abstraction Hierarchies Efficiently
76	Hierarchical data structures and algorithms for computer graphics
77	Hierarchical Path Planning for Multi-Size Agents in Heterogeneous Environments
78	Hierarchical routing for large networks
79	Identifying Hierarchies for fast optimal search
80	Implementation of parallel path finding in a shared memory architecture
81	Implementation of path planning using genetic algorithms on mobile robots
82	Improved heuristics for optimal path-finding on game maps
83	Improving collaborative pathfinding using map abstraction
84	Improving jump point search
85	Improving on near - optimality : more techniques for building navigation meshes
86	Smart moves: Intelligent Pathfinding.
87	K nearest neighbor path queries based on road networks
88	Lazy theta*: Any-angle path planning and path length analysis in 3d
89	Lifelong Planning A*
90	Field d* path-finding on weighted triangulated and tetrahedral meshes
91	Map complexity measure based on relative hamming distance

92	MAPP: a Scalable Multi-Agent Path Planning Algorithm with Tractability and Completeness Guarantees
93	Measuring Map complexity
94	Memory-efficient abstractions for pathfinding
95	Monte-Carlo Planning for Pathfinding in Real-Time Strategy Games
96	Multi- agent pathfinding, unexplored and dynamic military environment using genetic algorithm
97	Multi-agent pathfinding system implemented on XNA
98	Multi-core scalable and efficient pathfinding with Parallel Ripple Search
99	Multiple sequence alignment using Anytime A*
100	Navigation mesh generation in configuration space
101	Near Optimal Hierarchical Path Finding
102	Non-optimal multi-agent pathfinding is solved (since 1984).
103	Online graph pruning for pathfinding on grid maps
104	Optimal and Efficient Path Planning for Partially-known Environments
105	Optimal path-finding algorithms
106	Optimizations of data structures, heuristics and algorithms for path-finding on maps
107	Parallel multi-agent path planning in dynamic environments for real-time applications
108	Real-time path planning for virtual agents in dynamic environments
109	Path planning on cellular nonlinear network using active wave computing technique
110	Pathfinding algorithm efficiency analysis in 2D grid
111	Pathfinding and collision avoidance in crowd simulation
112	Pathfinding Design Architecture
113	Pathfinding in computer games
114	Pathfinding in partially explored games environments

115	Pathfinding in strategy games and maze solving using A* search algorithm
116	Pathfinding- Using interpolation to improve path planning: the field D∗ algorithm
117	Pathfinding: Real-Time Heuristic Search for pathfinding in video games
118	Performance analysis of pathfinding algorithms based on map distribution
119	Planning as heuristic search," Artificial Intelligence
120	Planning in a hierarchy of abstraction spaces
121	Portal-based true-distance heuristics for path finding.
122	Assessing the variation of visual complexity in multi scale maps with clutter measures
123	Rapidly-Exploring Random Trees: A New Tool for Path Planning
124	Real time search in dynamic worlds
125	Real-time heuristic search
126	Biased cost pathfinding
127	Self adjusting heaps
128	Simple optimization techniques for A* based search
129	Reducing the search space for pathfinding in navigation meshes by using visibility test
130	Simulation of dynamic path planning for real-time vision-base robots
131	Automated path prediction for redirected walking using Nav meshes
132	Strategic team AI path plans: probabilistic pathfinding
133	Sub-goal graphs for optimal pathfinding in eight-neighbor grids
134	Tactical path finding in urban environments
135	Pathfinding using tactical information
136	TBA*: Time bounded A*
137	Terrain analysis in real-time strategy games
138	The compressed differential heuristic

139	The focused D* algorithm for real-time replanning
140	Angelic Hierarchical planning: Optimal and online algorithms
141	The increasing cost tree search for optimal multi-agent pathfinding
142	The quadtree and related hierarchical data structures
143	The secrets of parallel pathfinding on modern computer hardware
144	Ultra-fast Optimal Pathfinding without Runtime Search
145	Adaptive grids: an image-based approach to generate nav meshes
146	Using Interpolation to Improve Path Planning: The Field D* Algorithm
147	Utilizing pathfinding algorithm for secured path identification in situational crime prevention
148	Video game pathfinding and Improvements to Discrete search on Grid-based maps
149	Generalized best-first search strategies and the optimality of A*
150	Correction to a formal basis for the heuristic determination of minimum cost paths
151	Shortest path algorithms: an evaluation using real road networks

Appendix B

	Papers with issues	Papers with empirical evaluation	Theoretical papers
Paper Numbers	2,3,7,18,19,20,23,25,27,28,29,31,32,35,37,39 40,42,43,45,46,47,48,49,50,51,53,54,55,56,5 7,60,62,63,64,65,66,68,70,71,77,78,79,80,81, 82,84,86,87,88,89,90,91,92,93,95,96,97,98,9 9,103,104,106,107,108,109,110,111,114,11 5,116,118,119,120,122,124,125,126,127,12 8,129,130,131,132,133,134,135,137,139,14 0,141,142,143,144,145,146,147,148,149,151	4,8,9,11,12,1 4,15,17,21,3 3,34,36,38,4 1,52,58,69,7 2,74,75,83,8 5,94,101,105 ,112,117,121 ,136, 138	1,5,6,10,13, 16,21,22,2 4,26,30,44, 59,61,67,7 3,76,100,1 02,113, 123,150
Total	99	30	22

Table 7: Classification of reviewed papers

1. **Issues in experimental design**: It cover the papers with experiments using only one or two type of maps, three or less map size variations, two or less obstacle density variation and no obstacle distribution.
2. **Issues in data collection**: It cover papers which collected data from 3 or less types of map and variations or data collected does not provide direct evidence supporting their claims like data of time consumption indirectly pointing to less memory consumption, no data for memory consumption.
3. **Issues in data analysis**: The papers that only provide average mean, median results and did not provide standard deviation, variance of the data.

	Experimental Design Issues [1]	Data Collection Issues [2]	Data Analysis Issues [3]
Paper Numbers	3,7,18,19,20,23,25,27,28,29,31,32, 35,37,40,42,43,46,49,53,54,55,,56, 60,62,63,65,66,70,71,77,79,80,82,84,87,88,89,92,93,95,96,97,99,103,104,107,108,110,114,115,116,118,119,122,124,127,129,131,134,135,139,140,141,144,145,147,148,151	2,7,19,20,35,42,45,48,50,51,55,60, 64,68,71,77,79,82,86,87,90,91,97, 104,108,110,119,122,129,133,135,140,147	3,7,20,25,32,35,37,39,40,48,55,57, 63,66,79,81,86,91,97,103,107,111,118,122,126,130,131,141,143, 149
Sub-Total	68	32	30

Table 8: Sub Classification of papers with issues

VITA AUCTORIS

NAME:	Harinder Kaur Sidhu
PLACE OF BIRTH:	Bathinda, India
YEAR OF BIRTH:	1992
EDUCATION:	St. Kabir Convent Senior Secondary School, Bathinda, India 2010
	GRDIET, B.Tech Computer Science Bathinda India, 2015
	University of Windsor, M.Sc. Computer Science, Windsor, ON, 2019