Application of Traveling Salesman Problem in Generating a Collision-Free Tool Path in Drilling

by

Zahra Khodabakhshi

A thesis submitted to the
School of Graduate and Postdoctoral Studies in partial
fulfillment of the requirements for the degree of

Master of Applied Science in Mechanical Engineering

Faculty of Engineering and Applied Science

University of Ontario Institute of Technology (Ontario Tech University)

Oshawa, Ontario, Canada

April 2021

THESIS EXAMINATION INFORMATION

Submitted by: **Zahra Khodabakhshi**

Master of Applied Science in **Mechanical Engineering**

Thesis title: Application of Traveling Salesman Problem in Generating a Collision-Free Tool Path in Drilling

An oral defense of this thesis took place on April 20, 2021 in front of the following examining committee:

Examining Committee:

Chair of Examining Committee	Dr. Martin Agelin-Chaab
Research Supervisor	Dr. Sayyed Ali Hosseini
Examining Committee Member	Dr. Amirkianoosh Kiani
Thesis Examiner	Dr. Ghaus Rizvi

The above committee determined that the thesis is acceptable in form and content and that a satisfactory knowledge of the field covered by the thesis was demonstrated by the candidate during an oral examination. A signed copy of the Certificate of Approval is available from the School of Graduate and Postdoctoral Studies.

ABSTRACT

In machining, the tool path is generated according to the workpiece geometry and arrangement of holes. Majority of Computer Aided Manufacturing (CAM) software offer a set of predefined strategies to choose from. These tool paths are mostly far from being the optimum path, specifically for complex geometries with non-flat surfaces. This thesis introduces a new algorithm based on Travelling Salesman Problem (TSP). The proposed local search algorithm generates an optimum collision free tool path in drilling operations. The developed optimization algorithm considers multiple constraints such as location of tool origin and presence of obstacles. Furthermore, a discussion on stopping criteria for the developed algorithm is presented. Obtained results confirm the proposed algorithm is capable of providing optimum collision free path with more than 50% reduction (in given examples) in path length compared to the HSMWorks software.

Keywords: Drilling; Tool path optimization; Collision-free tool path; TSP; Local Search method

AUTHOR’S DECLARATION

I hereby declare that this thesis consists of original work of which I have authored. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I authorize the University of Ontario Institute of Technology (Ontario Tech University) to lend this thesis to other institutions or individuals for the purpose of scholarly research. I further authorize University of Ontario Institute of Technology (Ontario Tech University) to reproduce this thesis by photocopying or by other means, in total or in part, at the request of other institutions or individuals for the purpose of scholarly research. I understand that my thesis will be made electronically available to the public.

Zahra Khodabakhshi

STATEMENT OF CONTRIBUTIONS

I performed the majority of the idea synthesis, development of algorithms, and writing of the following manuscripts.

Khodabakhshi, Z., Hosseini, A., & Ghandehariun, A., 2020, A Novel Method for Achieving Minimum Distance Collision-Free Tool Path for Drilling. *Proceedings of the Canadian Society for Mechanical Engineering International Congress 2020*.

Khodabakhshi, Z., Hosseini, A., 2020, Optimization Approaches for Minimizing Non-Productive Tool Path in Drilling, *17th IFAC Symposium on Information Control Problems in Manufacturing INCOM 2021*- In press.

DEDICATION

To myself for my hard work, devotion and perseverance;

To my loving mother, father, sisters and brother for their endless love;

To my husband, Mehran, for his support that cannot be expressed in words;

ACKNOWLEDGEMENTS

First and foremost, I would like to express my deepest gratitude to my supervisor Dr. Ali Hosseini for his invaluable guidance and full-heartedly support. I hugely thank him and I will always be proud of having such a big opportunity. I profoundly thank Dr. Kishawy for his valuable insights.

I would also like to thank other members of my lab over that past few years for their continual help. I would like to acknowledge the assistance and support from Ontario Tech University and in particular faculty of engineering and all its member's staff for all the considerate guidance.

My deep appreciation also goes out to all who have generously taken time to teach me new concepts. I am very fortunate to have all these great people in my life.

TABLE OF CONTENTS

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS AND SYMBOLS

CNC	Computer Numerical Control
CAD	Computer Aided Design
CAM	Computer Aided Manufacturing
TSP	Travelling Salesman Problem
MTPC	Multi-Tool hole drilling with Precedence Constraints
NN	Nearest Neighborhood
LS	Local Search
LK	Lin-Kernighan method
SA	Simulated Annealing
GA	Genetic Algorithms
ACO	Ant Colony algorithms
PSO	Particle Swarm Optimization
PCB	Printed Circuit Board
DE	Differential Evolution
AIS	Artificial Immune System
CSA	Clonal Selection Algorithm
WOA	Whale Optimization Algorithm
G-code	Geometric code
ALO	Ant Lion Optimizer
DA	Dragonfly Algorithm
MFO	Moth-flame Optimization
SCA	Sine Cosine Algorithm

ABC	Artificial Bee Colony algorithm
FA	Firefly Algorithm
TLBO	Teaching Learning Based Optimization
DP	Dynamic Programming
MLI	Modulated Light Intensity
ToF	Time of Flight
DAG	Directed Acyclic Graph
VRP	Vehicle Routing Problems
ssSKF	Single-solution Simulated Kalman Filter
PC-TSP	Precedence Constraints TSP
ST-TSP	Single Tool TSP
MT-TSP	Multi Tool TSP
NP-hard	Non-deterministic Polynomial-time
RSS	Range Sequential Search

Chapter 1: Introduction

1.1 Preamble

Manufacturing industry is facing rapid growth in today's competitive environment and it is a substantial contributor to the world economy. To survive in this fast-developing environment, manufacturing sector has always encouraged research, and innovations to meet the accelerated demand for productivity, quality, and environmental sustainability.

Among the manufacturing processes, machining is a fundamental process that has been widely adopted due to its flexibility and availability. However, machining processes are typically time consuming and wasteful of material. Thus, parameters such as time, cost, and quality that affect productivity and sustainability of machining processes must be thoroughly studied. In this context, innovative and efficient optimization models need to be developed and their effectiveness in real case industrial settings must be verified. Such models can be focused on optimizing tool path, optimizing machining parameters and also optimizing machine tools through better machine and tool design [1, 2]. Successful optimization models undoubtedly play a key role in achieving economic viability in machining industry.

Application of sensor integrated tools and machines, smart machine tools, and intelligent five-axis Computer Numerical Control (CNC) systems are examples of machine tools design evolution and optimization. Many parameters are involved in the machining process, such as spindle speed, feed rate, depth of cut, etc., which affects the process outputs like metal removal rate, tool life, surface finish, cutting forces and cutting time. Implementing optimization techniques for finding a satisfactory combination of machining and tool parameters is the main focus of in optimization of machining parameters.

This thesis mainly focused on tool path optimization. Tool path is the motion of cutting tool during machining process that eventually generates the desired geometry on the workpiece. Tool motion or tool path can be productive or non-productive [3]. In productive movements, metal cutting takes place due to the engagement of tool and workpiece and chips are formed and removed [4, 5]. Non-productive movement is a movement in which no cutting action occurs and there is no engagement between the tool and workpiece. This motion brings the tool to the desired position/location and thus used for positioning. Non-productive movement is also known as airtime motion [6]. It has also been referred to as non-functional trajectories [7, 8]. Optimizing productive and non-productive movement of the tool is the focal point of tool path optimization.

However, optimizing machining processes is not a straightforward task. For instance, optimizing machining parameters and machine tools is constrained by limitations of machine tools in terms of feed, velocity, and acceleration along with technological aspects of machine tool itself. Also, because machining parameters are highly dependent on the tool material, workpiece material, and industrial standards, changing them may alter the setting and jeopardize the smooth movement [9, 10]. As a result, tool path optimization in machining is very popular and has been the focus of many research works [11, 12]. Tool path optimization is also vital for improving and upholding machining productivity and quality [13].

Among machining operations, drilling is one of the widely used and well-known ones. Almost 95% of the machined parts have holes [14] and thus must undergo drilling during their manufacturing sequence. In drilling, tool movement only relocates the tool between the desired locations of the holes to be drilled. Thus, airtime motion of tool in drilling has no effect on the production of the part and its geometry and thus is a good candidate for minimization of airtime. This is in contrast with milling in which the tool path directly involved in creating and thus constrained by the desired shape of the workpiece,

1.2 Research Motivation

Non-productive time during drilling, associated with repositioning and switching of the drill bit during the operation, i.e. airtime motion, is reported to constitute up to 70 % of the total processing time. Hence, minimization of non-productive time or any improvement in the tool path geometry can significantly reduce the machining time and cost, particularly in mass production or production of complex parts [2, 4]. With the introduction of Computer Aided Design (CAD) and Computer Aided Manufacturing (CAM) software, both productive and non-productive tool paths in drilling are automatically generated according to the workpiece geometry and arrangement of holes. The generated tool path is then converted to G-code to be executed by CNC machines [15]. According to Lazoglu, et al. [16] , Kiani, et al. [17] and Hajad, et al. [18], CAM software usually generates tool paths (manually programmed and automatically generated) only based on the geometric computations of the workpiece. Therefore, the generated tool path is generally not optimal. Several research works in the field of drilling tool path optimization limit their focus to only simple shapes and hole arrangements [19-22]. Many industrial products such as engine blocks, dies and molds, etc.; however, have complex geometries [23]. The complex geometries need extra caution when generating tool path to avoid any collisions between the tool and workpiece, which is a major concern of high-speed multi-axis machines [24]. Thus, the importance of studying techniques that can analytically achieve both optimized and collision-free tool paths is clear.

Despite its importance, optimization works simultaneously considering both minimum tool path length and no collision as constraints are very limited. In pursuit of this idea, this thesis aims to address the shortcomings of the available research through presenting a new optimization model considering aforementioned constraints (minimum path length and no collisions with obstacles) for drilling processes. This research only considers the drilling process while the logic of the developed model can also be applied to other processes like turret punching or those of continuous tool paths such as milling and laser cutting [13, 23].

1.3 Thesis scope and outline

The thesis consists of four main sections. The first step is focused on investigating the theoretical background needed to understand the approaches used in the collision free tool path optimization area. This includes a thorough review of the current works considering their methodologies, optimization model, solution procedure and, finally selecting a proper model and an algorithm for solving the problem. In the next step, the problem and the selected algorithm will be discussed in detail. The algorithm will then be customized to the special constraints of the drilling process defined in this thesis and will be implemented for different scenarios.

In the third step, the algorithm will be verified and the results will be compared to those of CAM software to verify the performance and validate the proposed model. The scenarios will also be examined in different aspects for tackling the manufactures and the customers' special needs, and the results will be discussed in detail for decision making processes.

The fourth and final step of this thesis summarizes the improvements to the process proposed and discusses the road map for future works.

Chapter 2: Literature review

2.1 Preamble

In this chapter, key topics including drilling process, tool path, and Travelling Salesman Problem will be explained in detail. Then a thorough review of the published literature in the field of tool path optimization with the focus on drilling process and collision avoidance will be discussed. Finally, the finding in six main categories will be presented and summarized.

2.2 Definition of tool path

Regardless of the process, movement of tools in traditional machining processes like drilling, milling and turning and non traditional processes like water jet or electric discharge machining is either productive or non-productive [3]. The productive movements occur when material removal takes place and chips are formed [4, 5].

In contrast, non-productive movements take place when tool or machine head moves but no material is removed. Non-productive movements are mainly used for positioning and are also referred to as airtime motion or non-functional trajectories [6-8]. In this case the cutting energy (E_{cut}) is zero since there is no load on the cutting tool [25].

2.3 Importance of tool path Optimization

Turning, milling and drilling operations are the most widely used metal removal processes in which tool path optimization has already been thoroughly studied. Productive tool path in such processes directly governs the part geometry. Particularly, in milling and turning, cutting tool is engaged in cutting and removes material as it passes over the workpiece surface; thus, tool path cannot be altered without consequent effect on the part geometry. In drilling, however, cutting tool only removes material at desired location of holes and it is not engaged in cutting when moving from one location to another. Thus, its motion from point to point has no effect on the final part geometry. As a result, when it comes to tool path optimization, drilling is a great candidate. Non-productive tool path in drilling is associated with repositioning of the drill bit during the operation, i.e. airtime motion. Airtime motion is estimated to be up to 70 % of the total processing time; therefore, considerable efforts have been invested in minimizing airtime in drilling [2, 26, 27].

Nowadays, tool path in machining is usually generated by CAM software, however, the generated tool path is not necessarily optimum with minimum airtime [28]. It has been shown that the tool path generated by CAM software is generally not optimal from the optimization viewpoint and its efficiency highly depends on the user's experience and expertise [18, 29]. Lazoglu, et al. [16] emphasized that CAM software generates tool path mainly based on the geometric computations of the workpiece. Other papers highlighted the fact that majority of CAM software offer a set of predefined drilling strategies or a list of built-in modules to choose from [14, 16-18, 20, 27] . Other researchers aimed at optimizing tool path and compared their work with CAM generated one and they showed the path generated by CAM software is not optimum [14, 20]. Lee, et al. [30] stated that commercial CAD/CAM systems are somewhat incapable of satisfying manufacturers' needs and they do not allow users to apply field rules.

Almost none of the reviewed papers mentioned the name of CAM software they studied and they only use the term CAM software. Pezer [20], however studied three different CAM software namely WinCAM, CAMConcept and CATIA V5 and proved their inability in creating an optimum tool path. Also, none of the papers reviewed described how CAM systems generate the tool path. Only one paper mentioned that tool path in CAD/CAM software is generated based on the nearest-neighbor heuristic algorithm [18]. Their claim could not be validated since the commercial software of choice was not identified. It is evident that both manually programmed and automatically generated tool path by CAD/CAM software do not necessarily offer the optimum tool path with minimum overall distance to be travelled.

2.4 Tool path generation

Tool path (both productive and non-productive) for very simple scenarios can be generated manually. In complex cases, the tool path is generated automatically using CAD/CAM software and the user can just select a proffered tool path strategy from a set of predefined paths. The suitability of choice mainly depends on the user's decision [10, 28]. The preferred tool path strategy is selected by the user based on the following parameters [28]:

- Workpiece shape and geometry
- Workpiece material and microstructure (although particularly microstructure change during cutting process and may vary at different points in the workpiece)
- Parameters such as depth of cut, chip width and velocity (they are relatively easy to calculate based on the workpiece geometry and tool motion)
- Tool geometry
- Tool properties such as material and coating (if any)

Considering all the known parameters, the process engineer will then [28]:

- Select application of cutting fluid if needed as well as type and method
- Select tool path (both productive and non-productive) and CAD/CAM software helps in generating tool path.
- Visualizing the tool path for interference checking

Once the tool path is accepted, a cutter location (CL) file is generated by the CAD system and the postprocessors generate command to be executed by machine tool [5, 24]. As a result, both productive and non-productive tool paths in drilling are usually generated by modern CAD/CAM in the form of G-code according to the workpiece geometry and arrangement of holes [15].

2.5 Hole drilling process

In hole drilling process, a drill bit is used to cut a circular cross-section hole in the workpiece. Drilling may require a single tool where all holes to be drilled are similar (see Figure 2.1) or may need multiple tools where holes of different diameters must be created. In multi tool drilling, holes of similar diameter are grouped together and assigned to the appropriate drill bit (see Figure 2.2).

The ultimate scenario happens when each hole requires a predetermined sequence of drilling processes or drill bits. For instance, a hole needs to be predrilled, widened, and then finished by a taping or reaming operation. It is important to mention that these sequences are specified beforehand. This case is referred to as multi-tool hole drilling with precedence constraints (MTPC) [2]. As can be seen in Figure 2.3, the hole with a large diameter must initially be predrilled using drill bit 1. This hole is then further widened by drill bit 2, and ultimately sized to the desired diameter by drill bit 3.

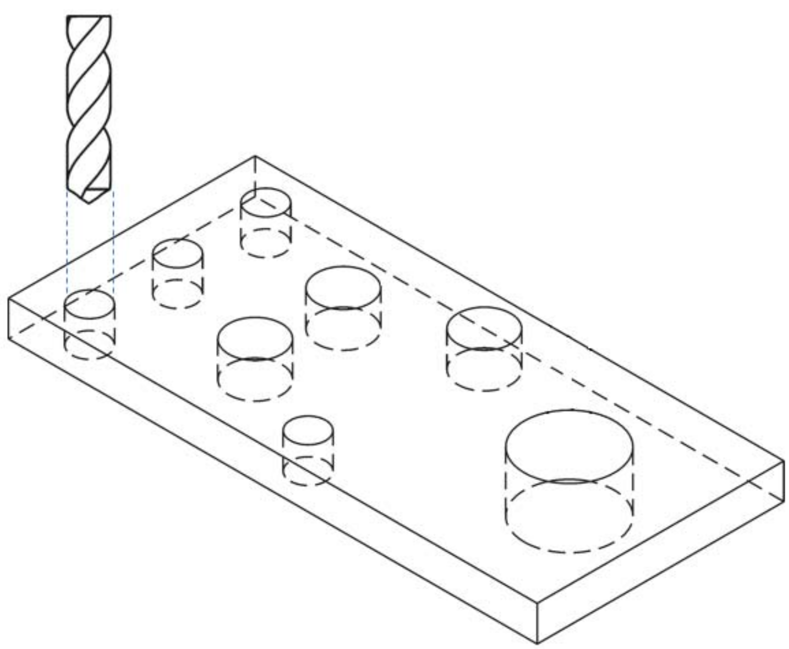


Figure 2.1: Single tool hole drilling workpiece

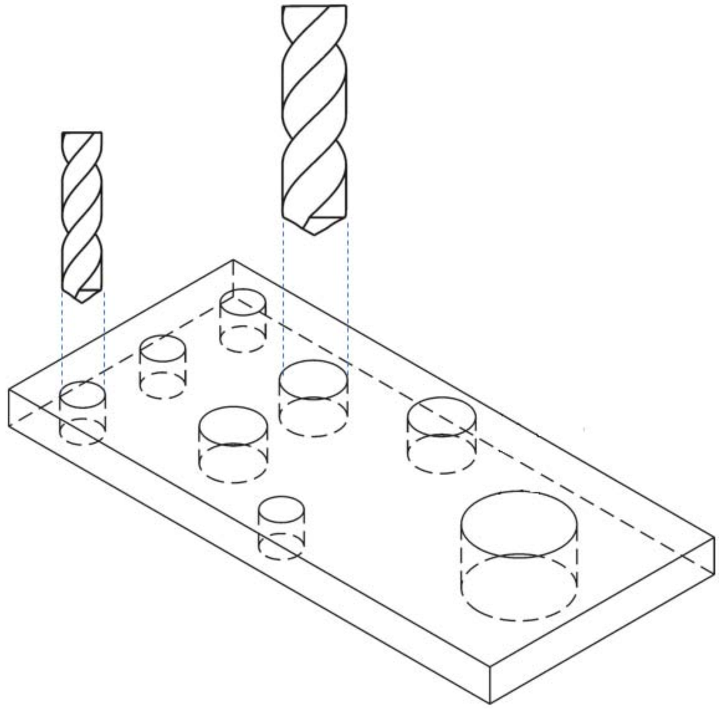


Figure 2.2: Multi tool hole drilling workpiece

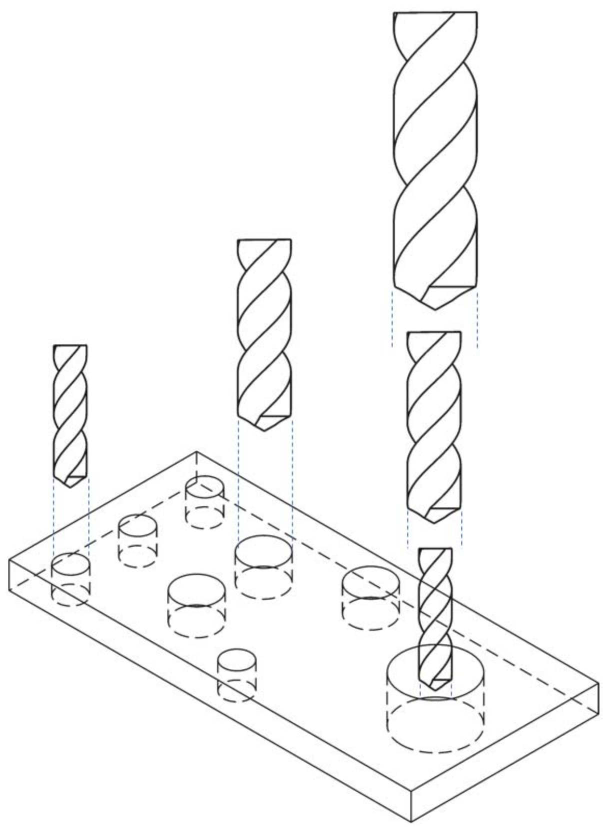


Figure 2.3: Multi tool hole drilling with known operations

2.6 Traveling Salesman Problem (TSP)

2.6.1 Origin of the problem

Travelling salesman problem is a popular optimization problem based on a scenario at which a salesman leaves his town, tries to visit other cities that are listed for him, and returns home at the end. Travelling salesman can visit cities in different orders (Figure 2.4). However, among numerous possible combinations, only one is optimum with minimum travelling distance. Traveling salesman problem (TSP) is believed to have originated in the United States [31].

Researches cannot say exactly when this problem first came into use and its mathematical path is still obscure. Practically speaking, due to evidence found, cave people solved small versions of TSP for hunting and gathering with no doubt [31].

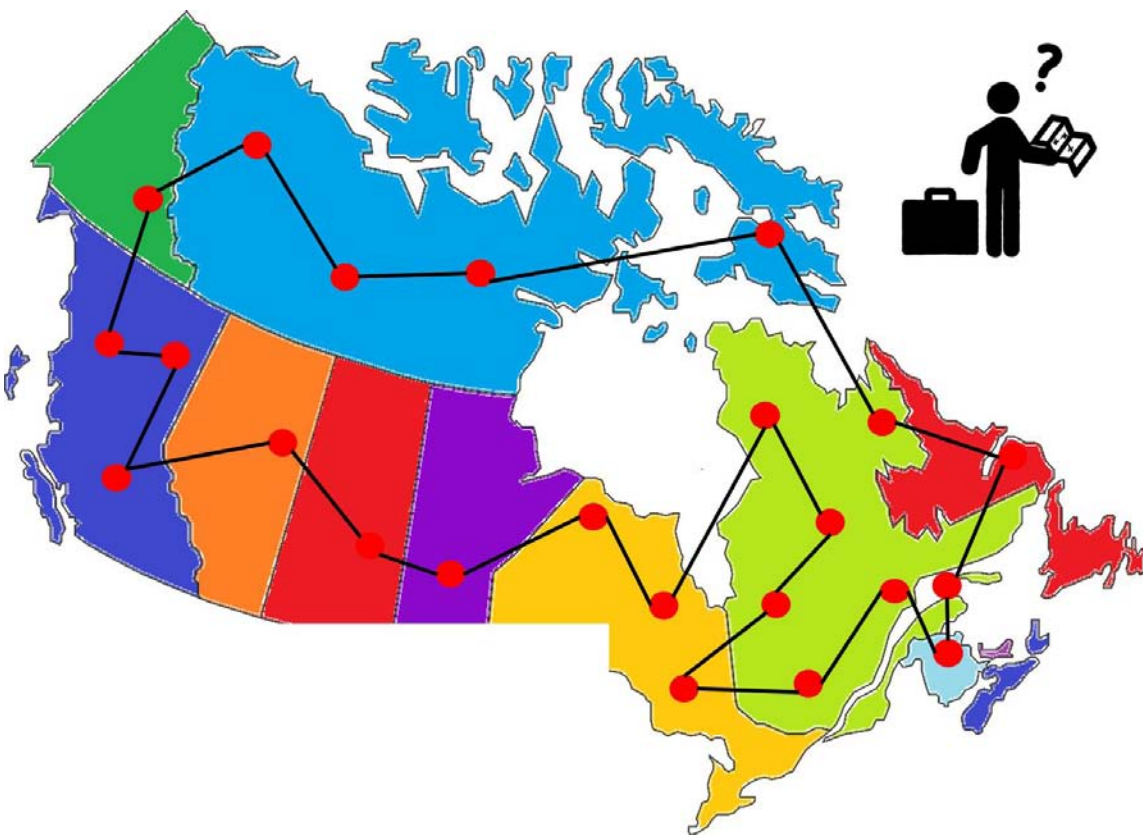


Figure 2.4: A salesman and a tour

Route planners were main users of this problem. In this discussion, an important reference is the 1832 German handbook [32]. The other example is the application of TSP in Page Seed Company by H. M. Cleveland in the year 1925 [31].

In 1930, Karl Menger, an Austrian mathematician and economist, brought the challenge of the TSP to the attention of the mathematics community for the first time [31]. In 1962, a contest with a $10,000 prize stimulated creativity among mathematicians. Two police officers, Toody and Muldoon, from a popular American television series, want to drive and visit 33 locations and travel the shortest possible route. Among all the people, two mathematicians, Robert Karg and Gerald Thompson produced the winning solution [31].

2.6.2 Why is TSP applied in drilling?

The problem of minimizing the path length between the holes during drilling or finding the best sequence of holes that are to be drilled can be described as a Travelling Salesman Problem (TSP) [23, 26]. A salesman in TSP must visit n cities with the condition that each city must be visited exactly once, and the salesman must return to the starting city. The final goal of TSP is to find the optimum path with minimum total traveled distance. As such, a similarity between the tool path optimization and the TSP can be directly devised. The cities are the holes to be drilled with the purpose of minimizing airtime and increasing productivity. One may believe that TSP is merely theoretical; however, it is a flexible yet effective method in solving several real-world applications. For instance, applications in logistics and transportation, which are the most common, planning, scheduling, and manufacturing even in machining [11, 33].

Our review of the literature confirmed that the TSP concept has been widely implemented as an efficient strategy for sequencing problems in the field of drilling process. According to Abidin, more than 90% of researchers applied this concept for generating an optimized path in drilling [23].

2.6.3 Approaches for solving TSP

Considering the complex nature of TSP problems, many methods have been developed in recent decades to solve this problem and new methods are still presented. In general, there are three ways for solving TSP, which can be divided into exact, heuristics and metaheuristics methods [20].

2.6.3.1 Exact Approaches

Exact approaches return the global optimum solution of the problem by solving all combinations of a problem to select the minimum distance combination. One of the most popular algorithms for finding exact solutions to TSP (discrete set of numbers) is a branch-and bound procedure. The simplest search strategy in branch and bound stands for creating all possible tours/configurations σ, and then calculating their corresponding distance values or objective function values $f(d)$. Finally, the path with minimum distance d_{final} is returned as the result of this search.

In this method, a search tree commences at a root node (starting city), then divides into branches (next possible cities), until the tree ends in the single leaves (starting city) [34]. Although exact algorithms can be very effective at solving instances of TSP with a very small number of cities involved, they usually fail when the problem sets become very large. To avoid the deficiencies of exact approaches in solving complex TSP problems and reduce computational time, heuristics and metaheuristics approaches are being used [2, 33].

2.6.3.2 Heuristics approaches

Heuristic approaches generate some possible combinations (solutions) instead of generating all possible combinations. According to Schneider and Kirkpatrick [34], heuristics approaches cannot provide a mathematical proof that the final combination is exactly optimal or at least how good the solution is compared to the exact optimal solution, but they could even be optimal if the number of holes/cities are limited. Heuristic approaches can simply be constructed in the programming software and they offer short processing time. However, these approaches are prone to trap in local optimum, which to some extent, hinders their effectiveness [35]. The most well-known heuristic methods are

Nearest Neighborhood (NN), General Local Search/Local Search (LS), 2-opt, 3-opt (K-opt), and Lin-Kernighan method (LK).

2.6.3.3 Metaheuristics approaches

In the last two decades, metaheuristic approaches have been increasingly proposed. Metaheuristic approaches are methods that provide good solutions to the proposed problems, which may not be attainable by the underlying heuristics approaches alone. Simulated Annealing (SA), Genetic Algorithms (GA), Tabu search, Ant Colony algorithms (ACO) and Particle Swarm Optimization (PSO) are the main categories of metaheuristics approaches. Like heuristics, metaheuristics approaches cannot assure that the final combination is exactly optimal or at least how good the solution is compared to the exact optimal solution. None of these methods can guarantee to find the exact/global optimum [34, 35].

2.7 Tool path optimization in drilling

In an attempt to optimize tool path in drilling, Kentli and Alkaya [36] applied a modified local search to solve a single tool TSP with 10-bolt assembly, 14-hole drilling and 442-point Printed Circuit Board (PCB) drilling problem. For comparison purposes, the same problems from previous literature were used. Comparison results showed that the proposed model was able to generate a considerably better solution in small problems while in problems with more holes, performance improved only by 3%. As a result, the proposed approach gave an acceptable solution to engineering problems especially in smaller scales.

Aciu and Ciocarlie [37] applied Lin Kernighan-Helsgaun (LKH) algorithm for PCB drilling. They used three PCB with 257, 481 and 985 holes. For G-code generation they used PCB-gcode-3.6.0.4 plugin and a User Language Program (ULP). Results demonstrated a 70% reduction in the tool path length compared to the G-code generator software for all three PCBs. The polynomial time complexity of the TSP was also demonstrated in this research. While the number of holes doubled, the computational time increased almost sevenfold; however, the total execution time of the LKH algorithm was still perfectly feasible, being around 182s for a PCB with 985 holes (see Figure 2.5).

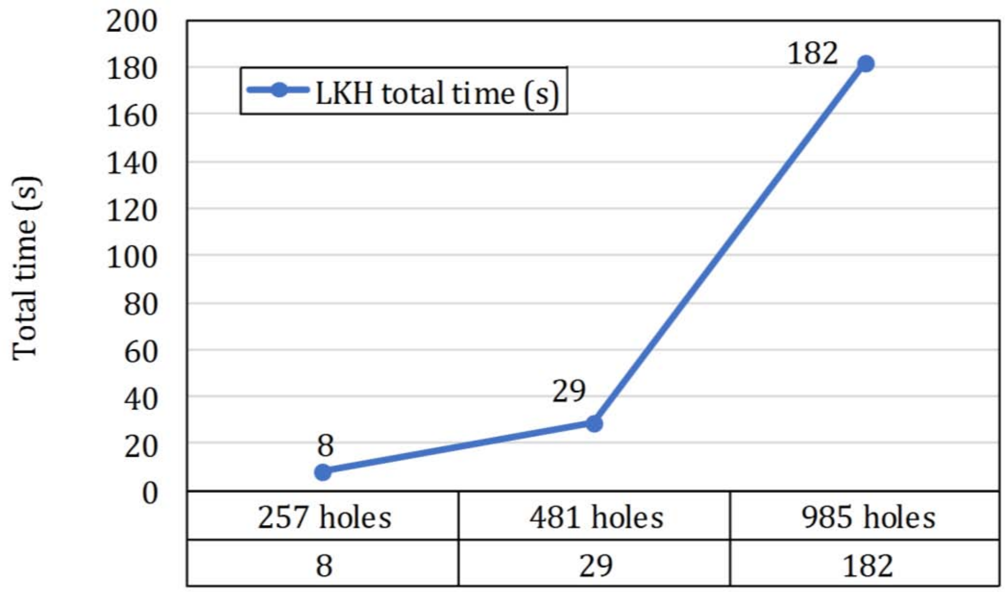


Figure 2.5: LKH total computational time (data from [37])

It was found that multi tool problems can be treated similar to single tool drilling problems. According to these research's findings, approaches for solving multi tool drilling optimization problems can be categorized into two groups [2, 11, 30].

In the first group, a small change can be made in the configuration of objective function matrix. In drilling problems with a single tool, the objective function matrix is the travel distance between any two holes. On the other hand, in basic multi tool problems, the objective function matrix consists of the distance between two holes i and j plus the distance that must be traveled to switch the tool for drilling hole j. As a result, from an optimization viewpoint, drilling problem with multiple tools reduces to the single tool problem. The only difference is earlier we defined a simple distance matrix that explicitly contained the travelling distance between one hole to another hole. Now, this distance (each element in the matrix) is equal to the summation of travel distance and tool switch distance.

As an example, Onwubolu [11] employed Differential Evolution (DE) for PCB CNC drilling. A CNC machine with two degrees of freedom in X and Y directions was used for drilling seven holes using four different drill bits. DE was solved with both forward and backward transformation techniques. The distance matrix was generated similar to the previously explained approach. The path was generated by applying DE algorithm to all holes while ignoring the difference in drill bits. Drill bits were then placed in the tool holder based on the optimum sequence generated in the previous step. The comparison between DE and other heuristic optimization techniques showed that the path length generated by DE algorithm was better.

The other approach to deal with multi tool drilling problems is grouping identical holes. In this approach holes with the same diameters are grouped. Each group is solved similar to single tool problems, then overall distance is calculated by adding the switch distances (the distance that each tool needs to travel to switch to another tool plus travel distance to the next group) to each group distance.

According to Lee, et al. [30], in the current system of machining marine engines, machining sequence is manually selected in the operation step, thus it requires many hours to create and edit the machining data . Thus, they applied the TSP model to find a proper drilling sequence for marine engine with three different tools. They grouped all the holes that needed similar tools together, namely group A holes with 30 mm diameter, group B counterbore holes with 20 mm diameter, and group C countersink holes with 30 mm diameters. TSP was then solved for all three groups. The increased efficiency of the proposed system was reported to be more than 60% in the actual industrial setting. No information regarding the TSP algorithm was provided. Hole groping approach used in [30] can be seen in Figure 2.6.

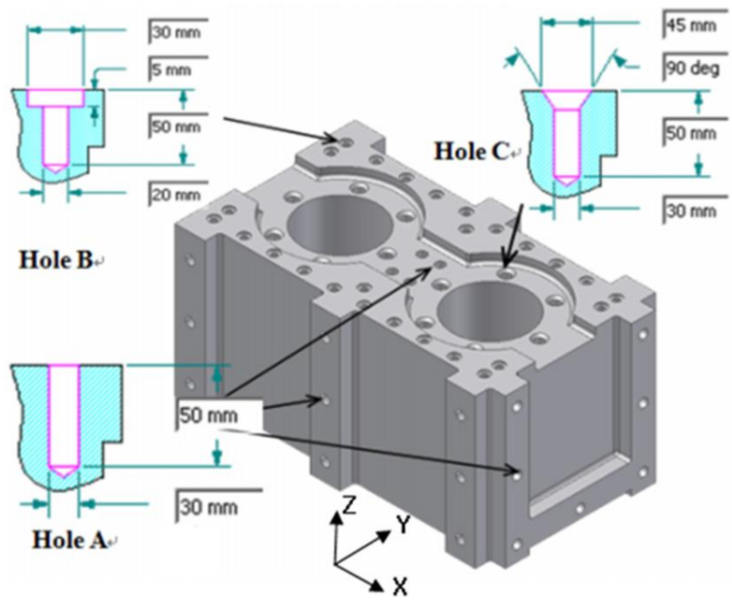


Figure 2.6: Engine Block and grouping holes based on tool (A, B and C) [30]

Huizar, et al. [15] solved the TSP problem with the Artificial Immune System (AIS). Clonal Selection Algorithm (CSA) as a common class of algorithms in AIS was used to decrease drilling time and cost by generating the optimal sequence of G-codes. Three experiments with different hole patterns and a single tool were performed. Each optimum path was then compared to the CAD/CAM obtained G-code path. The results showed CSA generated a significantly shorter path for drilling and manufacturing time was reduced by almost 35% to 53% according to the workpiece. Drilling path was a closed loop in which tool returned to the initial drilled hole. This closed loop method added extra distance to the overall drilling path. Common machining practice requires that the tool starts from a safe origin and returns to that origin at the end of the process [15].

Pezer [20] applied Genetic Algorithm on the principle of TSP to decrease tool path length in a prismatic workpiece. The results obtained from Matlab software were compared to CAM software (WinCAM, CAMConcept and CATIA V5). The total distance of tool path length achieved with CAM Concept, Win CAm and CATIA V5 programs were 1994 mm, 1017 mm, and 982 mm respectively, while the optimum path generated by GA was 869 mm obtained in 919 seconds run time. Genetic algorithm provided a better solution in relation to the all three software. Among the three software, CATIA solution was closer to the GA solution (see Figure 2.7). The computational run time by the GA was also investigated vs surges in number of iterations [20]. Higher number of iterations had a lower chance to be stuck in a local optimum, but the result would be obtained in significantly more computational time. It was concluded that by increasing the number of iterations, the quality of the obtained solution and computational time increased. The results obtained after 5000 iterations were accepted (869 mm), while the author showed that in 10000 iterations after 16 minutes, the model generated a better solution with the objective function value of 866 mm [20]. This emphasized the fact that we are forced to either accept the high computation times or a lower solution quality.

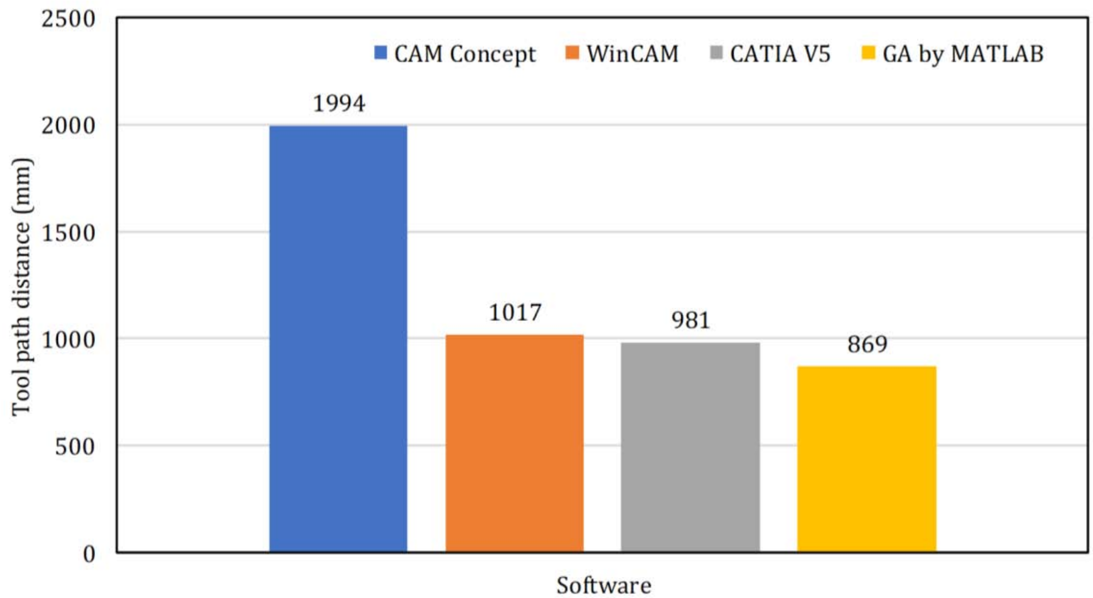


Figure 2.7: Results of tool path distance for various software and proposed GA (data from [20])

Narooei, et al. [38] applied ACO algorithm to generate the optimum path in drilling for a simple workpiece with 6 to 12 holes. They investigated the effects of control parameters (ρ, β, α) in ACO algorithm on the generated tool path. They observed that finding a suitable set of control parameters values affected the quality of the global solution, while they did not propose a method to find that suitable set. On the other hand, the distance function is according to the 2D Euclidean distance with a fixed z parameter equal to 1.5 cm (depth of holes is 1 cm plus 0.5 cm, which is the length between the tool tip and the workpiece surface).

Abidin, et al. [13] investigated the performance of PSO by comparing the results to GA, ACO, Whale Optimization Algorithm (WOA), Ant Lion Optimizer (ALO), Dragonfly Algorithm (DA), Moth-flame Optimization (MFO) and Sine Cosine Algorithm (SCA). Fifteen tests were performed ranging holes from 50 to 150 with maximum 300 iteration. Based on the observations the ACO algorithm performed better in small size problems mainly less than 50 while in larger numbers PSO algorithm showed better performance. Results indicated that new algorithms like WOA, ALO, DA, MFO, SCA were not suitable for discrete combinatorial optimization problems. For the reason that their final solutions were significantly larger and the computational time was higher (run time for each model is not mentioned in the article).

Six approaches namely: ACO, Artificial Bee Colony algorithm (ABC), PSO, Firefly Algorithm (FA), DE and Teaching Learning Based Optimization (TLBO) algorithm were applied to generate the optimal path in drilling in the scholarly work of Diyaley, et al. [14]. The results of these six algorithms were compared to the path that is generated by CAD/CAM software. The minimum path that is generated in all three tests (120, 250, 2600 holes) by all six optimization techniques proved to be shorter in length than CAD/CAM generated path. Amongst them, TLBO algorithm performed best with respect to the derived optimal path length and computational time.

Ghaiebi and Solimanpur [39], solved a precedence constraints TSP by ACO algorithm and LS in hole drilling. The initial population was generated by ACO and it was improved by local search algorithm. 12 holes and 6 tools were considered in their work. Their objective function consisted of tool airtime and tool switch time. The time simply calculated by dividing rectilinear distance function by the linear velocities in the x and y directions. Further into the article the velocities considered constant at $v_x = v_y = 1\, m/min$. For performance evaluation, proposed ACO was compared to a reference solution derived from Dynamic Programming (DP). In the performance step, in a range from 5 to 20 number of cities the proposed algorithm was able to generate hole drilling sequence close to DP in less computational time, however, from 25 to 50, the DP was not able to solve the problems in a reasonable time so the performance in this range was not investigated.

The similar formulation (objective function, distance, time…) and example to the work of Ghaiebi and Solimanpur [39], seen in the work of Lim, et al. [21]. They applied a hybrid Cuckoo Search - Genetic (CSGA) Algorithm for hole sequence optimization problem. It is proposed that CSGA performs well when compared to ACO, PSO, IAS, and cuckoo search alone. Each heuristic and metaheuristic algorithm have strength and drawback, Table 2.1 summarizes the advantages and disadvantages of some of the approaches mentioned in the reviewed papers. An overview of the reviewed papers is presented in Table 2.2.

Table 2.1: Advantages and disadvantages of common optimization techniques in the literature

	Advantages	**Disadvantages**
GA	Ability to efficiently explore the search space with randomization [21]	Selection of initial population highly affects optimum solution [20]
PSO	Ability to converge faster towards the optimal solution [13]	Extensive experimentation is required for initial setting of parameters [26]
ACO	Ability to solve the combinatorial optimization problems due to population-based optimization approach [38]	Selection of parameters highly affects the final Solution [38]
TLBO	Satisfactory performance due to involvement of less algorithm-specific parameters [14]	High computational time specially in complex discrete problems [19]
Cuckoo Search	Ability to find the desired solutions very efficiently for many continuous optimization [21]	For some examples an appropriate solution could not be found due to No Free Lunch theorem [21]

Table 2.2: Overview of reviewed literature

Reference	Year	Problem	Algorithm
[21]	2014	PC-TSP	Hybrid
[14]	2020	ST-TSP	ACO, ABC, PSO, FA, DE, TLBO
[27]	2011	ST-TSP	Modified ACO
[20]	2016	ST-TSP	GA
[29]	2011	MT-TSP	SA
[30]	2013	MT-TSP	Not mentioned
[26]	2004	ST-TSP	PSO
[36]	2009	ST-TSP	LS
[37]	2014	ST-TSP	LK
[11]	2004	MT-TSP	DE
[15]	2013	ST-TSP	CSA
[38]	2014	ST-TSP	ACO
[13]	2018	ST-TSP	PSO
[39]	2007	PC-TSP	ACO
[40]	2015	PC-TSP	GA
[41]	2009	ST-TSP	2-opt, LS
[42]	1998	ST-TSP	NN, SA, RSS
[22]	2008	ST-TSP	Modified LS
[43]	2017	ST-TSP	GA

RSS: Range Sequential Search, GA: genetic algorithm, ACO: Ant colony optimization, WOA: Whale Optimization Algorithm, ALO: Ant Lion Optimizer, DA: Dragonfly Algorithm, MFO: Moth-flame Optimization, SCA: Sine Cosine Algorithm, ABC: Artificial Bee Colony, FA: firefly algorithm, DE: Deferential evaluation, TLBO: teaching learning-based optimization, CSA: Clonal Selection Algorithm, PC-TSP: Precedence Constraints TSP, ST-TSP: Single tool TSP, MT-TSP: Multi Tool TSP.

2.8 Collision free tool path

A slight decrease in airtime path can significantly reduce the cost, however, this optimum path must be safe as well, especially during rapid displacement of tool in high-speed multi-axis machining environments [7]. Collision, if occurred, may damage the machine, workpiece, or both and leads to additional cost [24]. Detecting the possibility of collisions and avoiding them have many applications in industry. Collision detection and avoidance is also an important research field in other manufacturing areas like automated dimensional measurement inspection system [44] and robot path planning [45].

Literature review in the field of tool path optimization pertaining to drilling operation indicates that majority of research works have limited their focus on workpieces without any obstacle or geometric feature that prevents free movement of the cutting tool. Such assumptions may be valid for PCBs drilling or hole drilling of metal sheets [2, 10], while many other parts with real life applications have complex geometry and design. The importance of studying techniques that can analytically achieve optimized and collision-free tool path is clear. However, established literature regarding implementation of TPS to generate a drilling tool path with minimum length in presence of obstacles is very limited.

According to the work of Ahmad, et al. [8] limited effort has been done to create a collision free tool path. Collision is either prevented by the operator's intervention or predicted by CAM software during tool path generation. Although CAM software can detect the collision, they still leave the decision to the operator which in some cases leads to unexpected production stops. In Modern CAM software like Topsolid (Messler), the situation is still the same [8]. Sensors and vision-based methods are also used in preventing collision. Although they offer many advantages, their functionality may be jeopardised when their sensory capabilities or field of view is obscured by chips or cutting fluid [7, 45].

In the work of Senniappan Karuppusamy and Kang [10], a 2D top view CAD model for four workpieces was used for image processing. Initially, a 2D media filtering technique was applied to convert a top view color image of a workpiece into a grayscale image. This step was followed by noise removal and boundary tracing of the gray scale image by means of image processing techniques. Finally, black and white areas were presented as edge and machinable areas respectively in a 2D workpiece image. Then the image was divided into grids by an A* algorithm. In this algorithm each cell was given a weight to generate a cost matrix. If the tool was able to move in any direction, the cell got 2, otherwise the weight was zero. Finally, based on the generated cost matrix, GA was used to generate a near-optimal drilling path. The results of the proposed algorithm showed effectiveness, while, Ahmad and Plapper [7] reported some disadvantage of A* algorithm including leading to a local minimum and unacceptable space search of A* algorithm for many machine tools.

Visual based path planning using 2D and 3D cameras is another approach which has been studied in literature. Ahmad and Plapper [7] applied a Modulated Light Intensity (MLI) 3D sensor, also known as Time of Flight (ToF), to identify an imaginary polymer cube as an unknown obstacle during a non-functional tool path. Once the obstacle was identified by a 3D image from the sensor, the V-TRUST algorithm started to interpret the real time data to activate the predefined machine strategy to find a safe trajectory path for the machine tool to eliminate collision. Two strategies were defined to find safe points including above and in front of the obstacle [7]. The safe points were chosen according to the image and minimum distance between the tool and workpiece. However, this model assumed no chips and lubricant during image capturing which is rare in real world applications.

Sheng, et al. [44] addressed the path planning problem for a robot in a fully automated dimensional measurement inspection system. They considered the robot path as a TSP problem in a 2D plane. First both CAD and camera model was used to create the viewpoints, then the points considered as cities and solved with a modified NN algorithm. The mentioned algorithm ran for three automotive parts (door, floor pan and pillar). The results demonstrated a significant time saving for the mentioned robot while inspection.

Lee and Kim [45] used Directed Acyclic Graph (DAG) method for generating an initial population for GA for robot path planning. The objective was to find a path which starts from a point and ends in another point in the environment without intersecting any obstacles. They created a DAG that connected the starting point to the end point using nodes in the grid, and finally generated multiple paths based on the graph (see Figure 2.8).

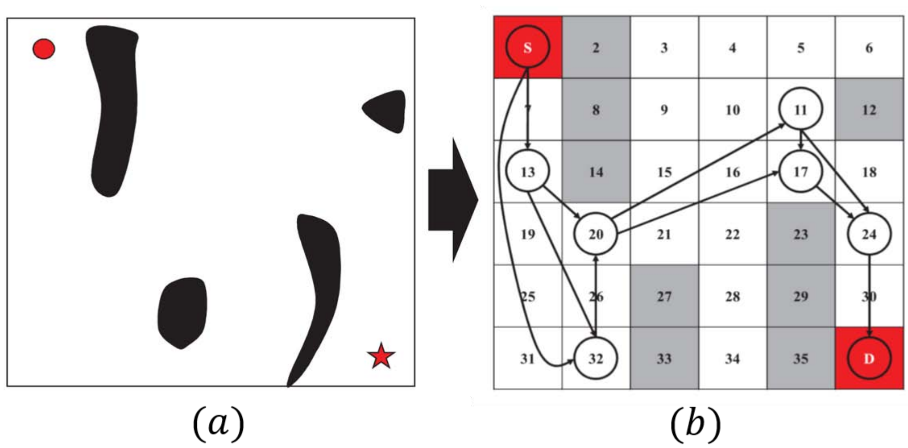


Figure 2.8: Proposed model steps based on [45] (a) example (b) DAG Algorithm

2.9 Discussion

2.9.1 Modelling approaches

Optimizing airtime path in drilling, includes minimizing the overall length that the drill bit travels, this can be described as a famous optimization problem called TSP. Figure 2.9 demonstrates the classification of the model itself, as can be seen, 85% of papers deal with TSP. Multi tool problems found in the reviewed papers are dealt exactly the same as single tool drilling problems, so these two scenarios merge together and refer to TSP. The remaining 15 % of implemented models deal with PC-TSP. PC-TSP is the common TSP with the restrictions that the drill bit should start from a predefined hole, e.g. a hole needs to be predrilled, widened, and then finished by a taping or reaming operation. PC-TSP is harder to solve due to the existence of sets of precedence constraints between holes. This area is not the focus of this research. To conclude, it is good to consider the fact that all problem types (Single Tool TSP, Multi Tool TSP, PC-TSP or Sequential Ordering Problem (SOP)) have the TSP-like nature.

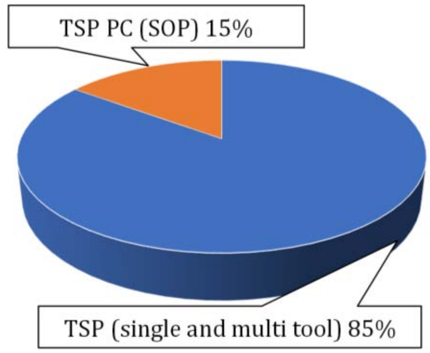


Figure 2.9: Overview of models used in hole drilling path optimization

2.9.2 Optimization Algorithms

Exact, heuristics and metaheuristics algorithms are used to solve TSP. Exact approaches have long computational time and they are incapable of generating a solution in complex problems. As a result, researches use heuristics and metaheuristics approaches to overcome the shortage of exact approaches and their long computation time. Figure 2.10 presents an overview of the algorithms used in the reviewed papers. As can be seen 73% of the applied algorithms are metaheuristics like SA, GA, PSO and ACO. GA and ACO have a great share of applied metaheuristic algorithms, while a small portion is dedicated to the new techniques like TLBO or FA. Heuristic approaches on the other hand gains 18% of the researcher's interest. One paper did not give any details on the algorithms used.

Almost half of the optimization approaches are population-based approaches, namely GA, PSO, CSA and ACO. The implantations of these classes of algorithms make sense since they can easily and quickly be applied to different types of TSP and PC-TSP. Once the population is created there is no other complexity required to check constraints especially in PC-TSP, this makes it easier for the user to work with these kinds of algorithms. The improvements in the programming software and hardware also have an increasing impact on implication of metaheuristic approaches. NN, local search and SA are kind of algorithms that need a good understanding of the problem and its neighborhood with extra effort in PC-TSP, so this makes sense that they are used only around 20%.

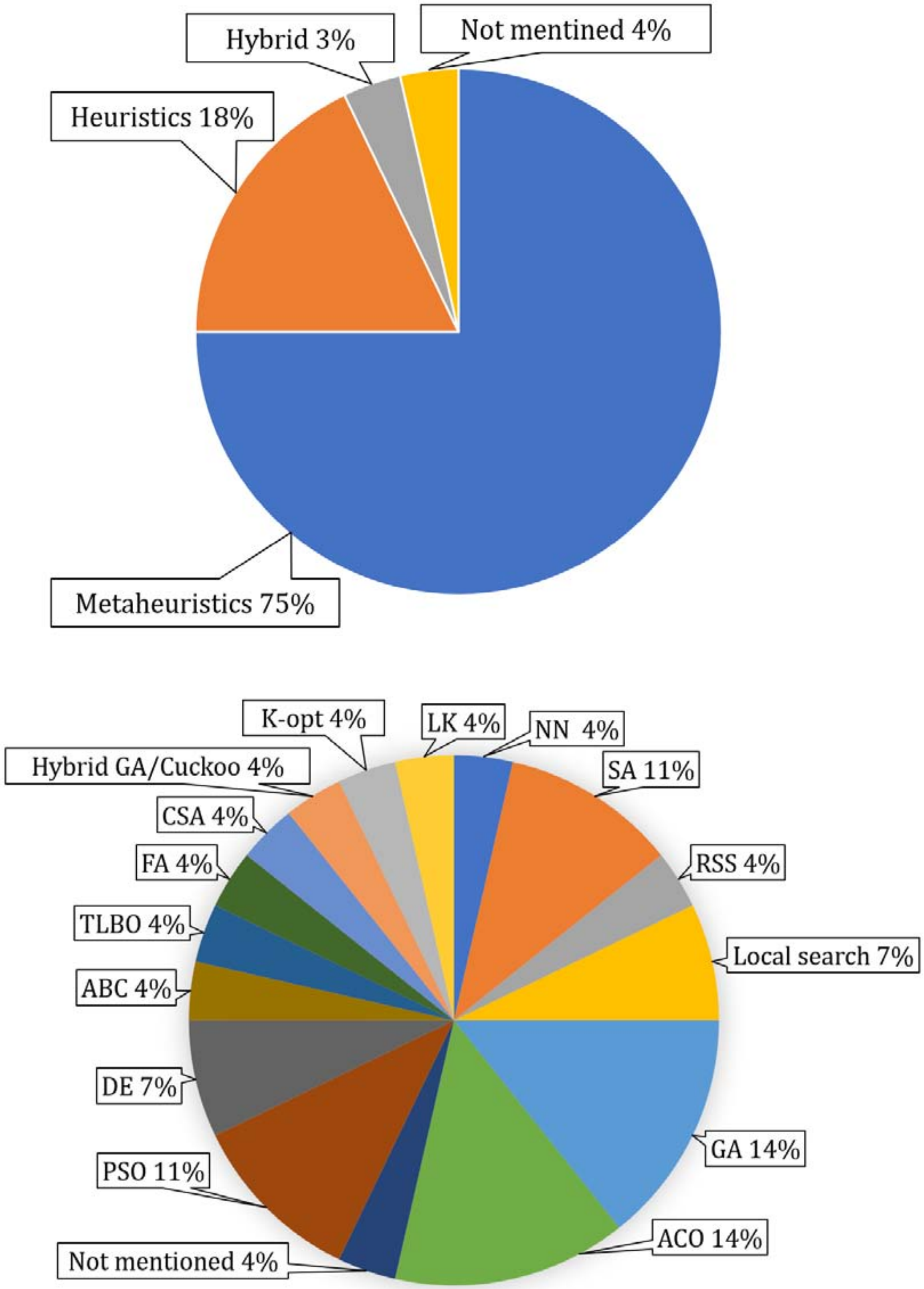


Figure 2.10: Overview of algorithms used in hole drilling path optimization

2.9.3 Application area

CNC drilling and PCB drillings are widely used in the reviewed papers for a better comparison these two main categories are separated. CNC machines improve productivity and quality especially on complex parts, since they are fully automated and require less manpower. PCBs are drilled with small-diameter drill bits which are typically made of solid coated tungsten carbide, and used in even the smallest electronic devices. PCB drilling usually made in large batch sizes from several hundreds to thousands of pieces. Figure 2.11 demonstrated that more than two-third of the papers are dedicated to CNC drilling only.

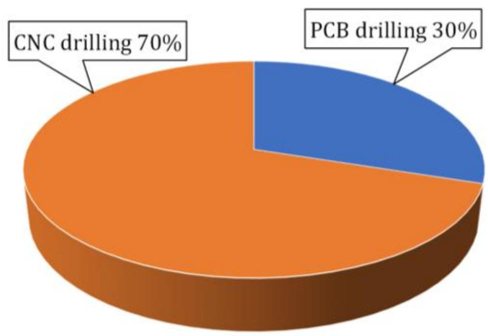


Figure 2.11: Overview of application area used in hole drilling path optimization

2.9.4 Objective functions

The objective function can be categorized into: minimizing the length or travel distance, reducing the drilling operation time and cost, and increasing productivity especially in PCB drilling by finding the optimal number of stacked PCBs to be drilled at the same time. Figure 2.12 below presented four objective functions that have been used in the papers.

The most frequently used objective is minimizing the distance. As can be seen, 65% of the reviewed papers used this objective function in drilling path optimization. The distance can be calculated using three different functions: Euclidean, Rectilinear, and Chebyshev. Among all, the Euclidean distance was used largely in the literature.

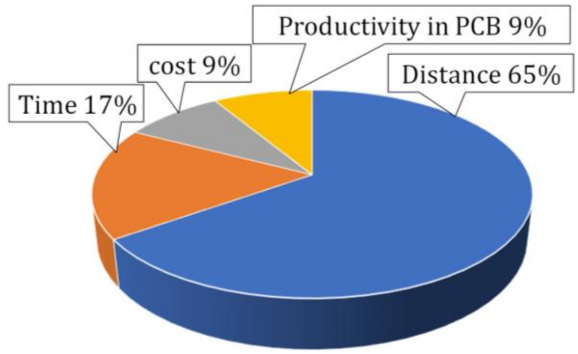


Figure 2.12: Overview of objective functions used in hole drilling path optimization

For minimizing time, which consists 17% of the reviewed papers, all is needed is dividing length of the path by velocity. In other words, the shorter the machining path becomes, the faster machining time will be. A constant velocity is assumed in all papers which is not a valid statement in field job. Machine tool head acceleration and deceleration is a significant factor especially when short distances are involved. Non-linearities aspect of velocities are not considered in the reviewed papers. This is understandable from an academic viewpoint, since all of the algorithms proposed used some kind of approximation to roughly calculate the travel time which is based on the total path length calculated. From the field viewpoint this matters greatly, and can be further discussed as an objective function (jerk) in future research field.

Cost is mentioned only in 9% of the works. It is calculated according to the relevant data for cost from the standard machining data handbooks, provided as machining cost per hole (productive cost) and non-productive cost per unit of length. As mentioned earlier PCBs are mainly produced in mass numbers, so increasing the number of productions each day is equal to a great productivity increase. 9% of the papers mentioned the subject of stacked PCBs, which means a number of PCBs lay on each other to be able to drill at the same time. As the number of stacked PCBs increase, the hole depth will increase as well and lead to an increase in drilling time. Consequently, drilling more PCBs in a stack does not necessarily lead to higher drilling operation productivity. To conclude, it is safe to say that in general, the main aim of all the objective functions are minimizing the distance while other parameters like time and cost will be calculated based on distance.

2.9.5 Returning to the initial city or tool safe origin

TSP is a common method of solving Vehicle Routing Problems (VRPs) in which a vehicle must start at a depot and distribute goods to a set of customers and return to the depot again for the next batch [46]. Therefore, almost all the reviewed papers applying TSP to the drilling process consider a closed loop for the tool path. This is in spite of the fact that, in real world manufacturing problems, returning to the first drilled hole is not required. Connecting the final city to the initial city adds an extra distance to the path which is not suitable for real world practice. This practice adds extra distance to the path as well as extra non-value-added time to the drilling process overall time.

This issue was addressed by solving the TSP problem with the closed loop assumption and then excluding the last distance from the final travelled distance [15, 20, 26, 47]. In the work of Zhang, et al. [47] this is called an open TSP. Excluding the last distance means, the tool stays in the last hole after drilling. This method is still not feasible in field work. In practical situations, the tool requires starting from a predefined origin and traveling through all the holes and returning to the origin position. Considering effects of tool origin in finding an optimum drilling path is not available in the reviewed papers. Huizar, et al. [15] referred to this issue as a future work. Effects of tool origin will be discussed in this thesis in the next chapter.

2.9.6 Computational time

Dealing with larger problem sizes or using metaheuristics over heuristics will cause computational time to increase. Here is the question, either accept the high computation time or to accept a lower solution quality. Whatever our selection is, we will end up sacrificing one of the aspects. In industry especially manufacturing, time is an important factor. Optimization aims to decrease the manufacturing time in order to survive in the competative world. Any fraction of reduction of time in machining processes matters a lot. While keeping this, another factor that can influence the computational time is stopping criterion. Selection of the stopping criterion depends essentially on the judgment of the user and often determined by the time and level of optimality. This will also be discussed in this thesis.

2.10 Summary

Although CAD/CAM software significantly helps to generate tool path, it is clear that both manual programming and an automatically generated path by CAD/CAM software do not consider the optimum method for creating a minimum overall distance. Therefore, the generated tool path is generally not optimal. To optimize hole drilling path, it is found 85% of the reviewed papers applied Traveling Salesman Problems (TSP) for which extremely powerful heuristics and metaheuristic approaches are used.

Reviewed papers in the field of drilling tool path optimization limited their work to workpieces without any obstacle or nonmachinable areas, while in industry parts have complex geometry and design. The importance of studying techniques that can analytically achieve optimized and collision-free tool path is clear. However, established literature regarding implementation of TPS to generate a drilling tool path with minimum length in presence of obstacles is very limited. In pursuit of this idea, this research aims to address the shortcomings of the available research through presenting a new TSP model with specified obstacles and constraints for drilling processes. This research only considers the drilling process while the logic of the developed model can also be applied to other processes with continuous tool paths, such as the milling process [23].

Chapter 3: Methodology

3.1 Preamble

Several researches in the field of drilling tool path optimization limit their focus to only simple shapes and hole arrangements without any obstacle, while real life industrial products have complex geometries [19-22]. The complex geometries need extra caution to avoid any collisions between the machine tool and workpiece features. In this chapter, TSP problem, its complexity and its mathematical formulation will be discussed, then the proposed algorithm will be presented in detail and finally the algorithm will be running for different scenarios. Straight walls and circular blocks are considered as obstacles in this thesis. A Personal Computer, with an Intel Core i5 processor at 3.1 GHz and 8 GB of RAM is used for all simulations. MATLAB R2019a software is used to run the proposed algorithm.

3.2 Mathematical Model of TSP

A salesman in this method has to visit n cities. Each city must be visited only once, and the salesman must return to the home city. In TSP, any sequence of all n cities that are visited by the salesman is called a tour. Similarly, any subsection of those n cities that still satisfy the definition (each city is visited once and salesman returns to the home city) is called a subtour [26, 48]. Figure 3.1 shows a typical TSP problem with 5 cities and some of its possible subtours and tours. As can be seen in Figure 3.1, the salesman has several options, more particularly $(n-1)!$ possibilities, to travel among the cities, visit each city once, and return to the starting city. The distance between each two consecutive cities (i, j) is represented by d_{ij}. For instance, the salesman can start from city 1 and travels through cities $2, 4, 3, 5$ and returns to city 1. The total distance of the tour will be $d_{12} + d_{24} + d_{43} + d_{35} + d_{51}$, see Figure 3.1(d).

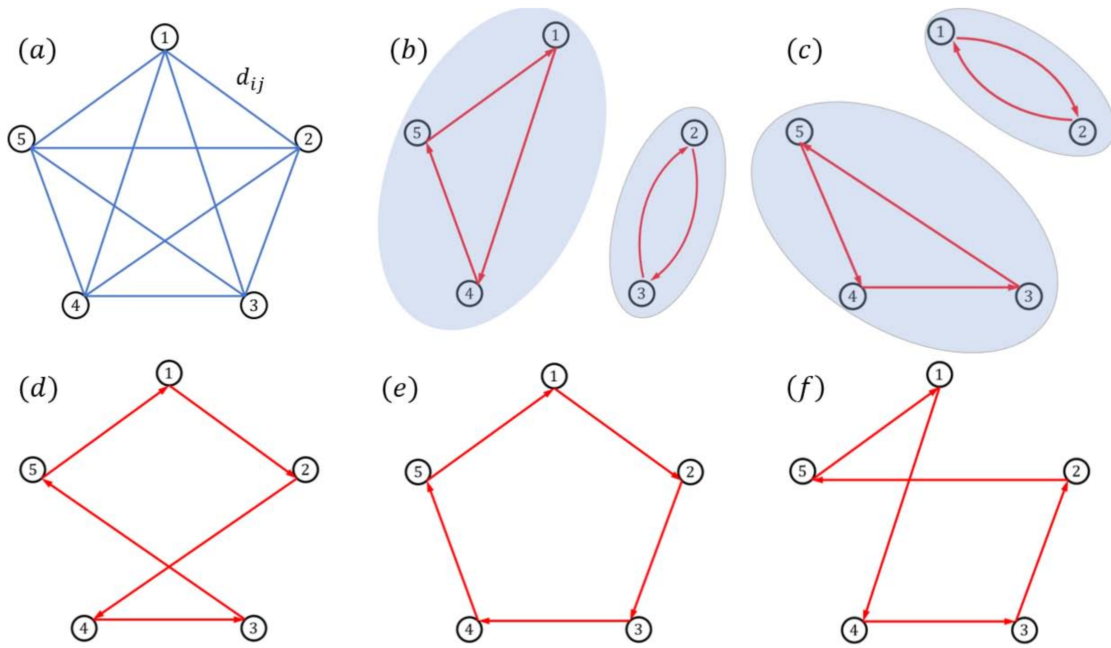


Figure 3.1: An example of TSP with five cities, (a) all possible paths, (b, c) two arbitrary subtours, and (d, e, f) three arbitrary complete tours

The subtours in Figure 3.1 (b, c) are also feasible solutions; however, since the concept of the tour is close to the concept of the subtour, many algorithms have been developed for subtour-elimination [48]. Individual looping (subtour) is not accepted in the original problem [6]. Various mathematical formulations can be used for solving the TSP problem. The common solution is to let K_{ij} be a decision variable which is defined as follows:

$$K_{ij} = \begin{cases} 1, \ if \ city \ j \ is \ immediately \ visided \ after \ city \ i \\ 0, otherwise \end{cases} \tag{3.1}$$

To describe the tool path, K is used as the decision variable where $K_{ij} = 1$ means that the salesman (or cutting tool in machining) travels from city (or hole in drilling) i to city j as a part of the final path. Similarly, $K_{ij} = 0$ means that the salesman (tool) does not travel from city i to j in the overall path [27]. Using this notation, the TSP problem can be stated as a minimization problem (see equation 3.2).

$$min\, Z = \sum_{i=1}^{n} \sum_{j=1}^{n} d_{ij} K_{ij} \qquad 3.2$$

Since the TSP goal is to minimize the total distance, the objective function described in equation 3.2 is to minimize the summation of all the distances d_{ij} in a tour (i.e. all possible combinations of tours without any subtours). The TSP problem is called Euclidean when the triangular inequality, as described in equation 3.3 is satisfied. d_{ij} refers to the Euclidean distance from city i to j [49].

$$d_{ij} \leq d_{ik} + d_{kj}\ for\ all\ (i, j, k) \qquad 3.3$$

The distance matrix D is defined as:

$$D = [d_{ij}], \forall\, i, j \in (1, \dots, n) \qquad 3.4$$

The tool path generation constraints can be mathematically formulated as follows:

(a) To ensure that each city j is visited only once in the tour [4, 12]

$$\sum_{i=1}^{n} k_{ij} = 1\, , \forall j \qquad 3.5$$

(b) To ensure that the tool leaves each city once [4, 12].

$$\sum_{j=1}^{n} k_{ij} = 1\, , \forall i \qquad 3.6$$

(c) To eliminate and disallow any subtour [12] (As mentioned above, no subtours means that there is no predefined priority for any of the cities and there is no need to return to or visit a city prior to the other cities [26]).

$$\forall S \subset \{1 \dots n\} : S = \emptyset \oplus \sum_{i \in S} \sum_{j \notin S} k_{ij} + k_{ji} \geq 2 \tag{3.7}$$

(d) To ensure that all cities are visited in a tour [12].

$$\sum_{i=1}^{n} k_{ij} > 0 \,, \forall j \tag{3.8}$$

(e) To ensure that every point is followed by a different point [12].

$$k_{ii} = 0, \forall j \tag{3.9}$$

(f) To investigate whether the TSP is symmetric, the following conditions must be checked:

$$\begin{gathered} if\ d_{ij} = d_{ji}\,, \forall (i,j) \\ \rightarrow the\ problem\ is\ symmetric; otherwise, it\ is\ asymmetric \end{gathered} \tag{3.10}$$

TSP can be symmetric or asymmetric. If the distances between each two cities differ depending on the movement direction, the formulation is asymmetric; otherwise it is symmetric [49]. In the drilling process, each node is determined by its x and y coordinates. Euclidean, Rectilinear, and Chebyshev distances between the cities i and j are calculated according to:

$$d_{euclidean,ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \tag{3.11}$$

$$d_{rectilinear,ij} = |x_i - x_j| + |y_i - y_j| \tag{3.12}$$

$$d_{chebyshev,ij} = Max(|x_i - x_j|, |y_i - y_j|) \tag{3.13}$$

3.3 Complexity of TSP

Visiting n cities might sound simple, however in reality this problem becomes more complex. The complexity of TSP is because of the fact that as the number of holes/cities increase, finding a solution becomes a Non-deterministic Polynomial-time problem (NP-hard problem). NP-hard problem means a difficult problem whose time complexity is exponential [35]. To describe more, the solution for the TSP problem lies in the possibility of finding the best/possible solution within a great number of possible combinations. The number of possible combinations in a symmetric TSP problem with n cities is:

$$Possible\ combinations\ in\ a\ symmetric\ TSP\ problem = (n-1)!/2 \qquad 3.14$$

If the problem is asymmetric the number of possible combinations is:

$$Possible\ combinations\ in\ an\ asymmetric\ TSP\ problem = (n-1)! \qquad 3.15$$

In order to further clarify this issue, reviewing an example of reference [31] is useful. Let's use $n = 33$, starting with a random city. 32 other cities are left for the second city, 31 choices for the third and etc. Overall, permutations of 32 cities (32!), $32 \times 31 \times 30 \times \cdot \cdot \cdot \times 3 \times 2 \times 1$ is considered. Considering symmetric assumption, so the 32! possible combinations can be cut down by half leaving only $32!/2$ combinations to check. Before you go ahead and get out your pencil for solving implicit simple TSP, note that

$$131{,}565{,}418{,}466{,}846{,}765{,}083{,}609{,}006{,}080{,}000{,}000$$

tours that need to be examined. One may say supper computers can be used to solve this problem. So, choosing the best one in Kobe Japan, Fugaku delivers up to minimum $442.010 \times 10\text{^}15$ Floating Operations per second (442 Peta Flops). Let's assume a single operation is needed to examine one tour. We would then need $9{,}437{,}304{,}489$ years, roughly 9 billion years, to solve a 33-city TSP. An unreasonable amount of time for solving a problem, given that the universe is estimated to be only 14 billion years old. According to equation 3.14 and equation 3.15 , if we increase the number of cities or a few number of elements, the number of possible combinations quickly gets out of hand [33] (see Table

3.1). It is also clear from Figure 3.2 that by increasing the number of cities, the number of possible combinations in both symmetric and asymmetric TSP problems increase exponentially.

Table 3.1: Number of cities and possible combinations

Number of cities	Possible combinations	
	Symmetric TSP	Asymmetric TSP
5	$(5-1)!/2 = 12$	$(5-1)! = 24$
10	$(10-1)!/2 = 1.8\ e + 5$	$(10-1)! = 3.6\ e + 5$
20	$(20-1)!/2 = 6.1\ e + 16$	$(20-1)! = 1.2\ e + 17$
40	$(40-1)!/2 = 1.0\ e + 46$	$(40-1)! = 2.0\ e + 46$
100	$(100-1)!/2 = 4.7\ e + 155$	$(100-1)! = 9.3\ e + 155$
200	$(200-1)!/2 = 1.9\ e + 372$	$(200-1)! = 3.9\ e + 372$
500	$(500-1)!/2 = 1.2\ e + 1131$	$(500-1)! = 2.4\ e + 1131$

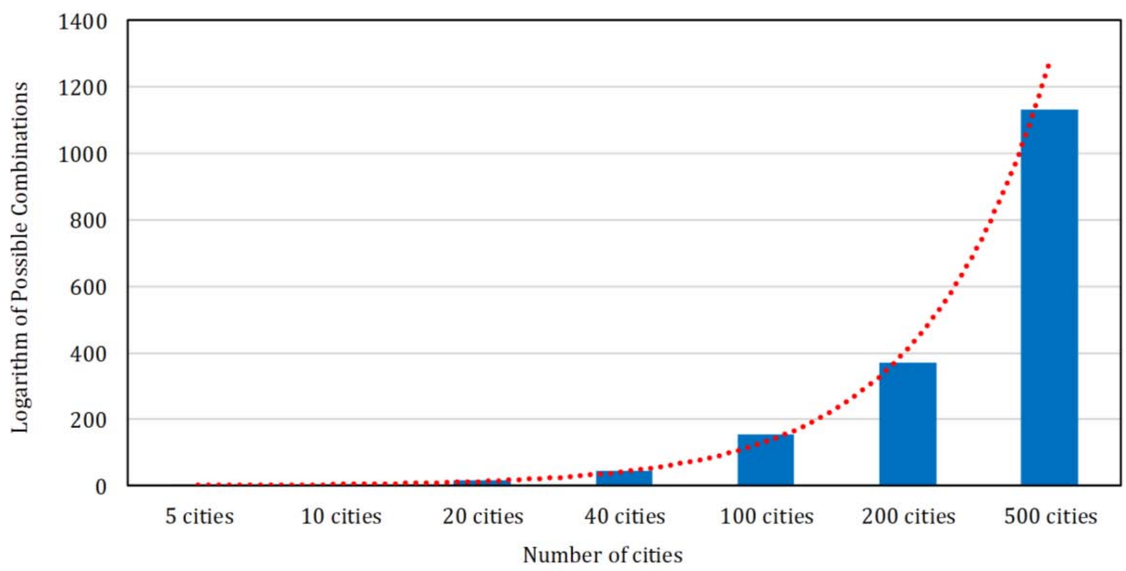


Figure 3.2: Number of cities and Possible combinations

In examples with few numbers of holes (namely less than 6 holes) it is still possible to generate all combinations and find the optimum solution, this determines a way of solving called exact approaches which will be discussed in detail in the following section. By a slight increase in the number of holes, a simple example with 10 holes, the possible combinations jump to 181,440.

It is impossible to generate all 181,440 combinations and find the best solution in a reasonable amount of time. This emphasises that TSP is an NP-hard problem. Therefore, algorithms used to solve TSP try to find a possible solution in a subset of all the possible combinations. No algorithms guarantee to discover global optimum for TSP in a polynomial time [27], but they can find a solution that is very close to the optimal in a reasonable amount of time [20]. Using heuristic and metaheuristic algorithms can give us good solutions in a timely manner, while sacrificing finding very good solutions in a polynomial time [50].

3.4 Heuristic algorithm: Nearest Neighborhood heuristic

The simplest idea to construct a tour is to travel to the closest city among those not yet visited. One of the famous heuristics for solving TSP problems is the Nearest Neighbor algorithm (NN). Some authors use the name greedy for nearest neighbor algorithm [51, 52].

Nearest neighborhood algorithms build tours by repeatedly choosing the closest eligible city until all cities are visited and the chosen cities form a tour. Nearest neighborhood is a constructive method. Constructive heuristics build a tour from scratch according to some construction rules and stops when a feasible solution has been generated [51, 53]. Table 3.2 shows the reasons for selecting the nearest neighborhood algorithm. According to the advantages presented in Table 3.2, and considering the vital role of time in manufacturing, the nearest neighborhood method is selected.

Table 3.2: Reasons for selecting nearest neighborhood algorithm

Advantage	• Have short running times compare to other approaches in both heuristic and metaheuristic domain • The relatively good results due to the its greedy nature • Ability to converge faster towards the near optimal solution due to the path extending in the shortest possible manner at each step • Ability to be served as a good starting tour to the metaheuristic approaches • The saved time in path generation step can be used in manufacturing [31, 53]
Disadvantage	• All heuristics algorithms have the possibility of getting stuck in local optimum • It looks very good for many steps but it does not search the overall neighborhood structure of the problem so all edges do not represent a short path [31, 53]

Nearest neighborhood algorithm obtains the following procedure:

Step 1. Start from an arbitrary start city. For $i = 1, 2, \ldots, the\ number\ of\ cities\ (n)$.

Step 2. Select another unselected city which is closest to the start city. for $j = 1, 2, \ldots, n-1$ that $d_{ij} = min\{d_{ij} | i \neq j\}$.

Step 3. Connect j to i. Algorithm will run until all the cities are visited.

Step 4. Choose a path to the first city in step 1 to form a complete and closed tour.

3.5 Proposed Nearest Neighborhood algorithm description

The TSP assumptions can be improved by adding new constraints for generating a collision free path. Additionally, the tool is assumed to start from a predefined origin position instead of a starting city in this thesis. Hence, the four corners of a workpiece are considered as possible initial tool positions. The best initial position is then selected in a way that the overall travel distance is minimized.

The initial algorithm used to solve the TSP drilling problem is the nearest neighborhood heuristic algorithm. The nearest neighborhood algorithm starts by selecting a starting city. The algorithm proceeds through $n-1$ stages, in each stage adding an unassigned city to the loop that is closest to the current city. Then the algorithm investigates whether the path to the next city has a collision with the obstacle. The sequence progresses by all remaining cities at each stage to meet all the constraints. The algorithm will be repeated each time with an initial selection of a different city. Finally, the near optimum path will be selected. The whole process will be performed for all workpiece corners to achieve the minimum path traveled by the tool. The computational steps for the application of the proposed model are defined as follows:

Step 1. Initialize from one/each corner of a workpiece and a start city. For $i = 1, 2, \ldots, the\ number\ of\ cities\ (n)$. Set it as the current city and mark it as visited.

Step 2. Select a new city from the distance matrix which establishes the minimum distance from the current city and mark it as the next city. for $j = 1, 2, \ldots, n-1$.

Step 3. Investigate whether the path to the next city has a collision with the obstacle. If "Yes" the tool will proceed to the nearest obstacle edge to avoid any collision and then select the next nearest city.

If "No" the algorithm will move to the next city and mark it as visited.

Step 4. Update the list of unassigned cities. Algorithm will run until all the cities are visited.

Step 5. Choose a path to the tool origin such that there is no obstacle on the path from the tool origin to the start city and the path from last city to the tool origin.

Step 6. In each corner, select the least overall travelling distance.

Step 7. For comparison purposes, distances from step 6 will be compared and the final path will be selected.

The flowchart for the algorithm is presented in Figure 3.3. As mentioned, the nearest neighborhood is a constructive method, i.e. the solution is found by adding components to a partial solution until the final solution is achieved. The workload will increase as the number of cities n increases [11].

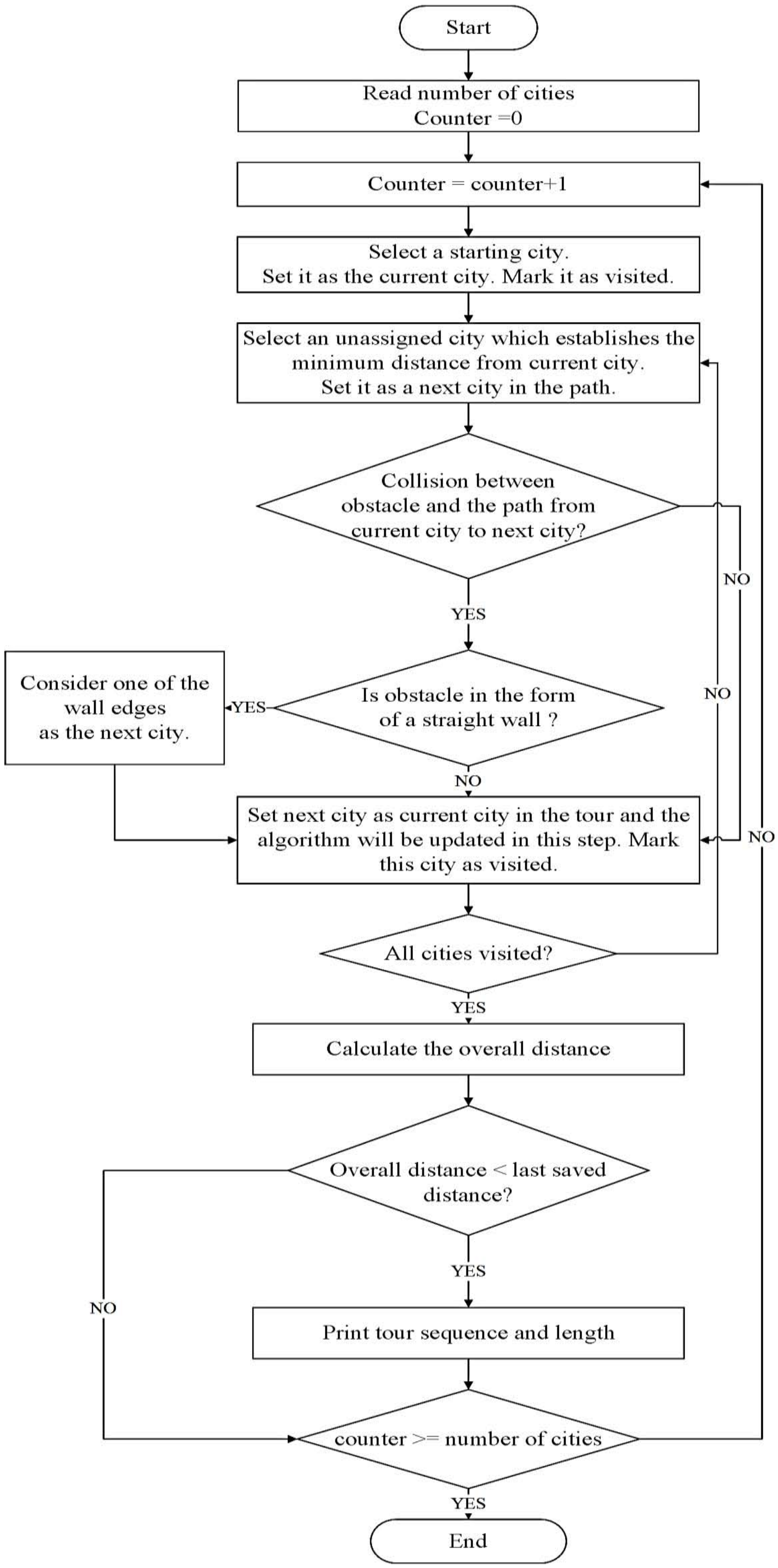
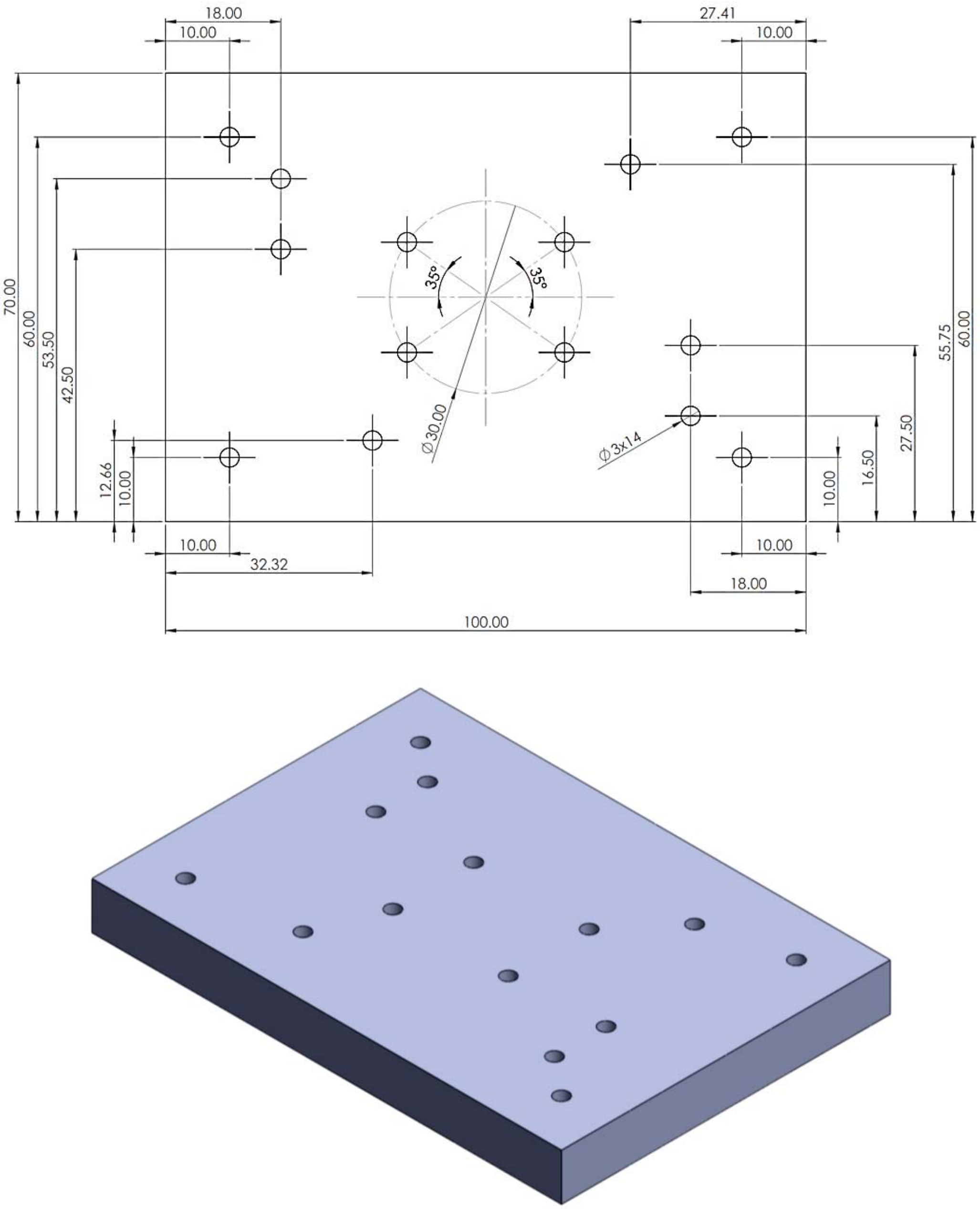


Figure 3.3: Proposed nearest neighborhood heuristic flowchart

3.5.1 Potential complexity#1: workpiece with two separate wall obstacles

In this section a widely used workpiece is studied. This workpiece is used in many works, some of which worth mentioning are Zhu [54],Zhu and Zhang [55], Aziz, et al. [56] and Kentli and Alkaya [36]. This single tool hole drilling workpiece with 14 holes is shown in Figure 3.4.

(a)



(b)

Figure 3.4: 14-hole drilling workpiece dimensions and arrangement of holes (a) 2D drawing; (b) Isometric view (all dimensions are in mm)

Table 3.3: Location of holes

No.	Hole coordinate (mm)													
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**
x	10	32.3	37.71	18	37.71	18	10	90	72.59	62.29	62.29	82	82	90
y	10	12.7	26.41	42.5	43.60	53.5	60	60	55.75	43.60	26.40	27.5	16.5	10

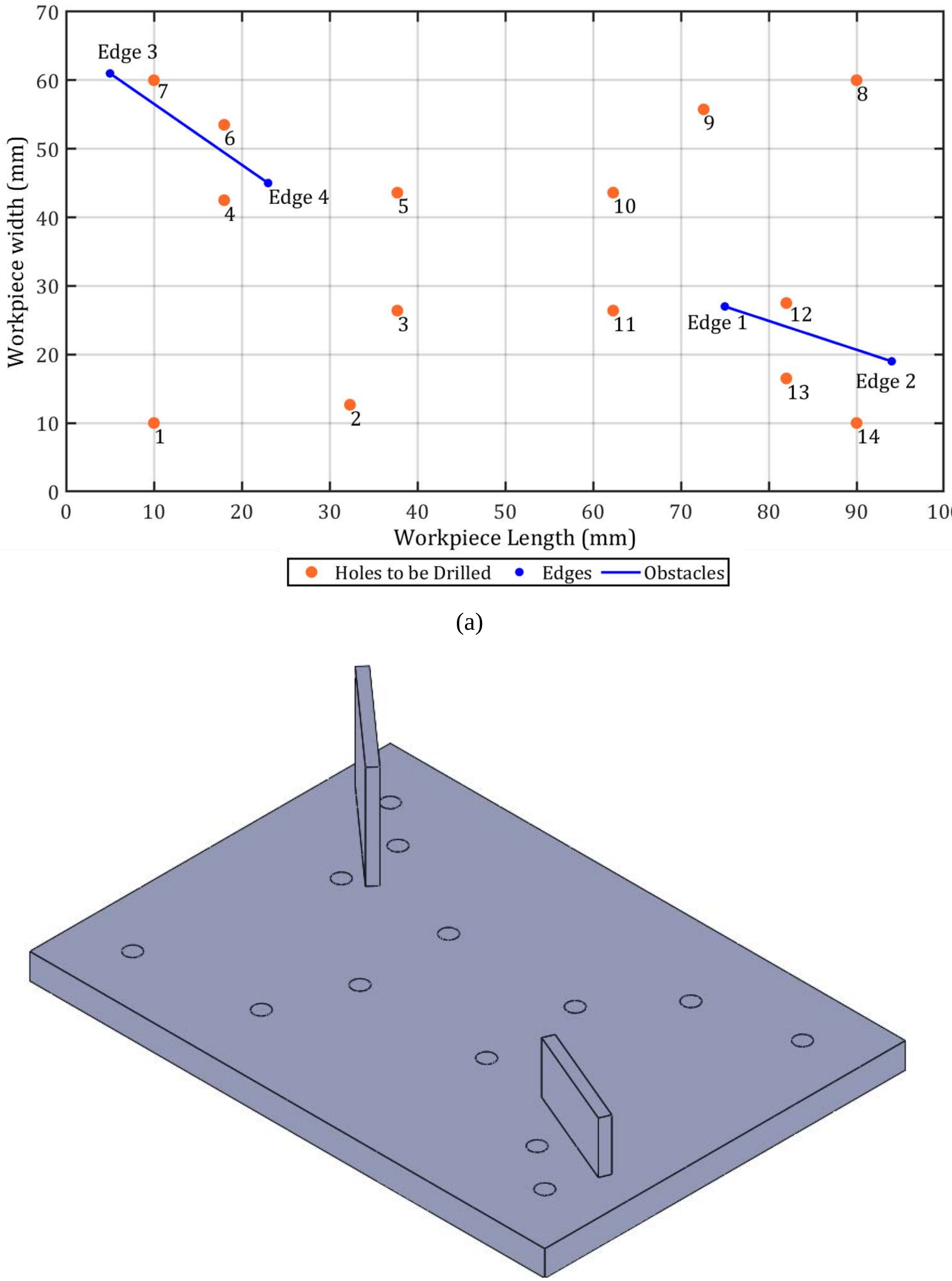
The depth and diameter of holes are considered consistent among all the holes. Therefore, the z travel distance for creating the holes is similar among all the holes and can be eliminated from the calculation. Likewise, no tool change is required during the process. As a result, the motion is considered 2D in the x and y directions.

The drilling path can be pictured as a TSP, where the salesman is the drill bit and holes are the cities. In order to understand the flow of the proposed algorithm, obstacles are added to the example shown in Figure 3.4. In this scenario, the tool is going to drill 14 holes on a workpiece with two straight wall obstacles. Height of the wall obstacles are 10 and 20 mm, respectively. Figure 3.5 shows dimensions of the workpiece with four corners (adjacent to four corners of the figure), location of 14 holes to be drilled (cities to be travelled), and arrangement of two wall obstacles. The obstacles force the tool to move around to avoid any collision. Table 3.4 shows the x, y coordinates of the wall obstacles.

Table 3.4: Location of edges

No.	Edge coordinate (mm)			
	1	**2**	**3**	**4**
x	75	94	5	23
y	27	19	61	45

For the problem presented in Figure 3.5, the proposed nearest neighborhood algorithm was executed for four different scenarios where the tool origin is located at (0, 0), (0, 70), (100, 70), and (100, 0), see Figure 3.6.



(a)

(b)

Figure 3.5: 14-hole drilling workpiece dimensions and arrangement of holes and obstacles, MATLAB figure (b) Isometric view (all dimensions are in mm)

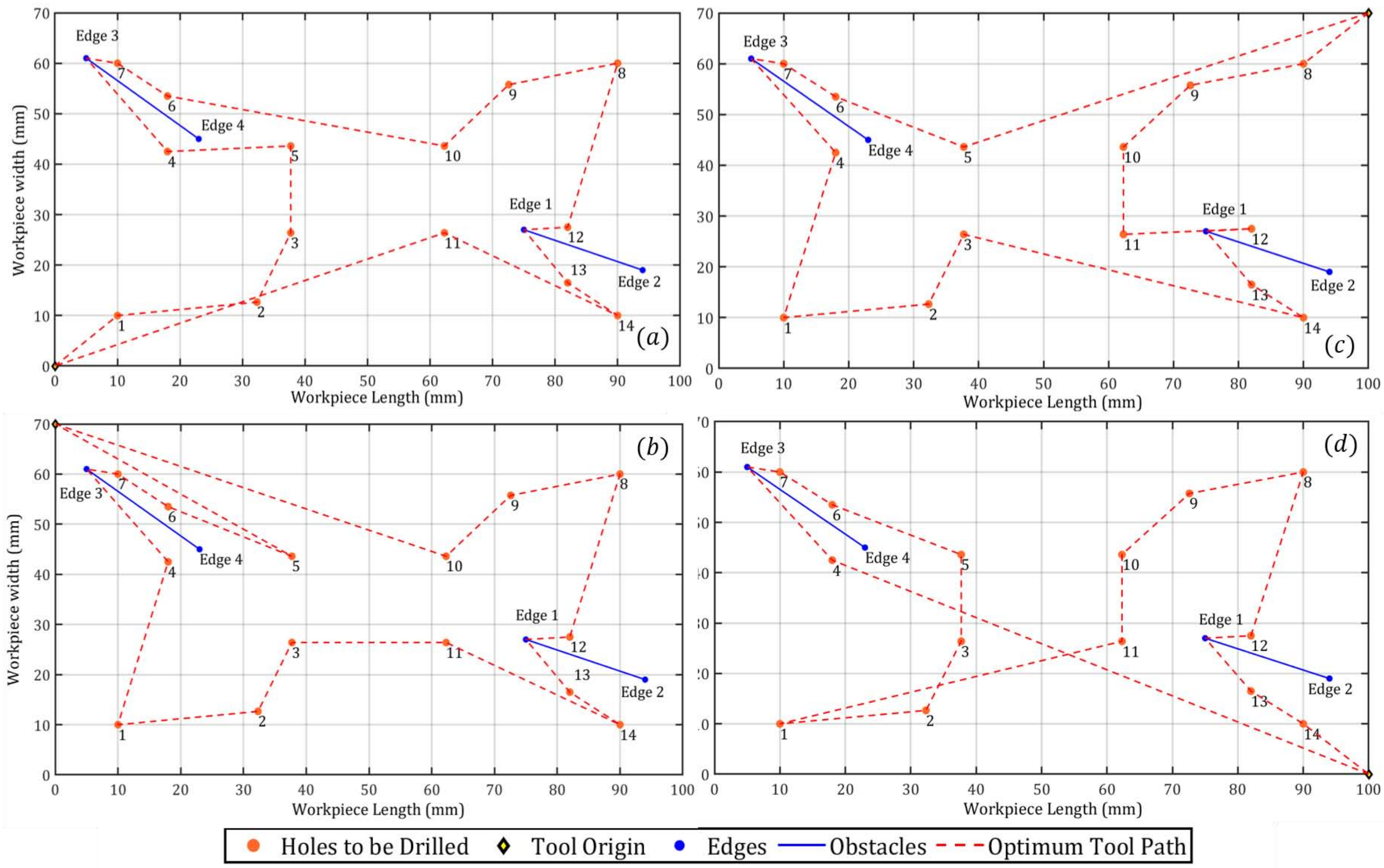


Figure 3.6: Near optimum tool path when tool origins is located at (a) point (0, 0), (b) point (0, 70), (c) point (100, 70), (d) point (100, 0)

As presented in Figure 3.6 (a), the near optimum TSP tour when the starting point of motion is located at (0, 0) is : tool origin (0, 0), hole 1, hole 2, hole 3, hole 5, hole 4, edge 3, hole 7, hole 6, hole 10, hole 9, hole 8, hole 12, edge 1, hole 13, hole 14, hole 11 and tool origin (0,0). For such a tour (considered also as the tool path), the objective function value (i.e. the near optimum length) is 368.84 mm. The total run time is 0.2 seconds. The remaining corners are also shown in Figure 3.6 (b), (c), (d).

The results are summarized in Table 3.5. For ease of tracing the optimum tool path, edge 1 is indexed as 15, edge 2 is indexed as 16, same for edge 3 and 4, and tool origin is indexed as 0.

Table 3.5: Summary of near optimum generated tool paths for different tool origins (case in Figure 3.5)

Tool origin	**(0,0)**	**(0,70)**	**(100,70)**	**(100,0)**
Near optimum path length (mm)	369	398	368	390
Computational time (seconds)	0.2	0.3	0.2	0.1
Path	$0 \rightarrow 1 \rightarrow 2 \rightarrow 3 \rightarrow 5 \rightarrow 4 \rightarrow 17 \rightarrow 7 \rightarrow 6 \rightarrow 10 \rightarrow 9 \rightarrow 8 \rightarrow 12 \rightarrow 15 \rightarrow 13 \rightarrow 14 \rightarrow 11 \rightarrow 0$	$0 \rightarrow 10 \rightarrow 9 \rightarrow 8 \rightarrow 12 \rightarrow 15 \rightarrow 13 \rightarrow 14 \rightarrow 11 \rightarrow 3 \rightarrow 2 \rightarrow 1 \rightarrow 4 \rightarrow 17 \rightarrow 7 \rightarrow 6 \rightarrow 5 \rightarrow 0$	$0 \rightarrow 8 \rightarrow 9 \rightarrow 10 \rightarrow 11 \rightarrow 12 \rightarrow 15 \rightarrow 13 \rightarrow 14 \rightarrow 3 \rightarrow 2 \rightarrow 1 \rightarrow 4 \rightarrow 17 \rightarrow 7 \rightarrow 6 \rightarrow 5 \rightarrow 0$	$0 \rightarrow 4 \rightarrow 17 \rightarrow 7 \rightarrow 6 \rightarrow 5 \rightarrow 3 \rightarrow 2 \rightarrow 1 \rightarrow 11 \rightarrow 10 \rightarrow 9 \rightarrow 8 \rightarrow 12 \rightarrow 15 \rightarrow 13 \rightarrow 14 \rightarrow 0$

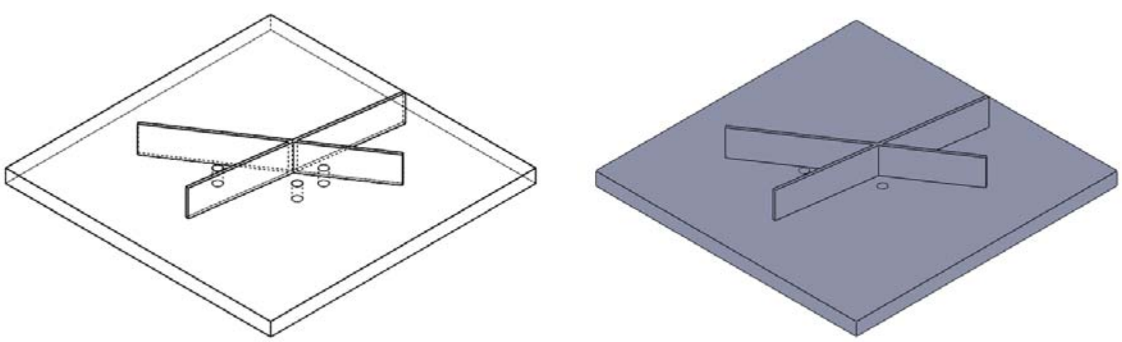
Regarding the results, the path starts from the top right corner shown in Figure 3.6 (c) has the minimum path length. Thus, the operator can define (100, 70) as the safe tool origin. Basically, in field work, safe tool origin is selected based on the operator's experience, so to reduce the intervention of the operator, this task can be fulfilled by the proposed model.

As can be seen, the optimum tool path length varies depending on the tool origin. As a reason, a good selection of safe tool origin will minimize the total drilling path length and save time especially in mass production. This issue was referred to a potential future work in the work of Huizar, et al. [15]. Computational time is acceptable for practical applications. Needless to mention that due to the nature of nearest neighborhood algorithm, the near optimum tool paths are local optimums. Global optimum for each case can be obtained by inspecting all possible combinations which is extremely time consuming; particularly, when the number of cities and obstacles increases.

3.5.2 Potential complexity#2: workpiece with two intersecting wall obstacles

The nearest neighborhood proposed algorithm has issues in solving problems with two or more intersecting obstacles. In such cases, the algorithm gets stuck in a loop and is unable to proceed forward. To further discuss the problem a workpiece with two intersecting obstacles is selected. Figure 3.7 shows a scenario where three holes must be drilled without colliding the walls or obstacles.

(a)



(b)

Figure 3.7: 3-hole drilling workpiece dimensions and arrangement of holes and obstacles, (a) MATLAB figure (b) Isometric views (all dimensions are in mm)

Table 3.6 shows dimensions of the workpiece, location of 3 holes to be drilled and arrangement of two colliding wall obstacles. Height of the wall obstacles are 10 mm. These obstacles can be geometric features of a workpiece in a real-life machining practice.

Table 3.6 : Location of holes and obstacles

No.	**Hole coordinate (mm)**			**Edge coordinate (mm)**			
	1	**2**	**3**	**1**	**2**	**3**	**4**
x	4	6	6	2	8	5	6
y	4	5	6	3	7	1.8	9

As can be seen, if the tool starts from hole 2, as an arbitrary starting point, the next nearest hole (regardless of the presence of obstacles) to visit is hole 3. Nevertheless, the direct path from hole 2 to hole 3 intersects the obstacle defined by edge 1 and edge 2. For ease of referring, obstacle defined by edge *i* and edge *j* is shown as $obstacle_{i\text{-}j}$. This obstacle must be cleared without any collision; thus, the algorithm identifies the nearest edge of that obstacle to the current tool position (hole 2), which is edge 2. The tool travels to edge 2 and then proceeds to hole 3. The last hole to visit (drilled) is hole 1. Similar to the previous step, the straight path from hole 3 to hole 1 initially collides with $obstacle_{3\text{-}4}$ and then with the $obstacle_{1\text{-}2}$. Therefore, the algorithm focuses on clearing the obstacles by traveling to its nearest edge to the current tool location which is edge 5. Travelling from edge 5 to hole 1, the tool now collides with $obstacle_{3\text{-}4}$ and the nearest edge of that obstacle to the current location of tool is edge 4; thus, tool will move to edge 4. If the tool travels from edge 4 to hole 1, it will again intersect $obstacle_{1\text{-}2}$ and the nearest edge of that obstacle to the current tool position is edge 2. Consequently, the algorithm is trapped in a loop between edges 2 and 4 (hole 2→ edge 2→ hole 3→ edge 2→ edge 4 → edge 2 → edge 4 → edge 2 → edge 4 …) as shown in Figure 3.8.

Figure 3.8: Inability of the Nearest Neighborhood in generating near optimum tool path, formation of loops on the path proves failure of this algorithm

In order to address this issue, two strategies can be implemented. The first strategy is to find the first potential collision point between the tool path and the obstacles, then to select that end of the obstacle which is closest to the current tool position. This strategy partially fixes the issue; however, it may fail in some particular occasions. In the same example, if the tool starts from hole 2, the next nearest hole to drill is hole 3. Nevertheless, a collision with an obstacle$_{1-2}$ will occur. This obstacle must be cleared without any collision; thus, the algorithm identifies the first potential collision, and then selects the nearest edge of that obstacle to the current tool position (hole 2), which is edge 2. The tool travels to edge 2 and then proceeds to hole 3. Now, the first strategy fails by travelling to hole 1 as the remaining hole to drill.

The path from hole 3 to hole 1 initially collides with obstacle$_{4-3}$ and then with the obstacle$_{1-2}$. Therefore, the algorithm focuses on clearing the first obstacle by traveling to its nearest edge to the current tool position (hole 3) which is edge 4. Travelling from edge 4 to hole 1, the tool now collides with the second obstacle$_{1-2}$ and the nearest edge of that obstacle to the current position of tool is edge 2; thus, tool will move to edge 2. If the tool travels from edge 2 to hole 1, it will again intersect with the obstacle$_{3\text{-}4}$ and the nearest edge of that obstacle to the current tool position is edge 4. Consequently, the algorithm is trapped in a loop between edges 4 and 2 (hole 2→ edge 2→ hole 3→ edge 4→ edge 2 → edge 4→ edge 2 → edge 4 → edge 2 …) as shown in Figure 3.9.

Figure 3.9: Inability of the first strategy in generating near optimum tool path, formation of loops on the path proves failure of this strategy

The second strategy selects the closest edge to the intersection point between the straight tool path and the corresponding obstacle instead of the closest edge to the current tool position. This strategy solves the previous issue as described below.

Starting from hole 2, the next nearest hole to drill is hole 3. A collision occurs with obstacle$_{1\text{-}2}$. The closest edge to the intersection is edge 2. The tool travels to edge 2 and then proceeds to hole 3. The path from hole 3 to hole 1 collides with obstacle$_{3\text{-}4}$ and obstacle$_{1\text{-}2}$. Then the first intersection is selected and the closet edge to this intersection point will be edge 4. Travelling from edge 4 to hole 1, the tool now collides with the second obstacle$_{1\text{-}2}$, based on this strategy the next tool position will be edge 1 and then finally hole 1 (see Figure 3.10). However, it is efficient only for simple problems but fails to generate a solution for more complex ones such as the one presented in Figure 3.11.

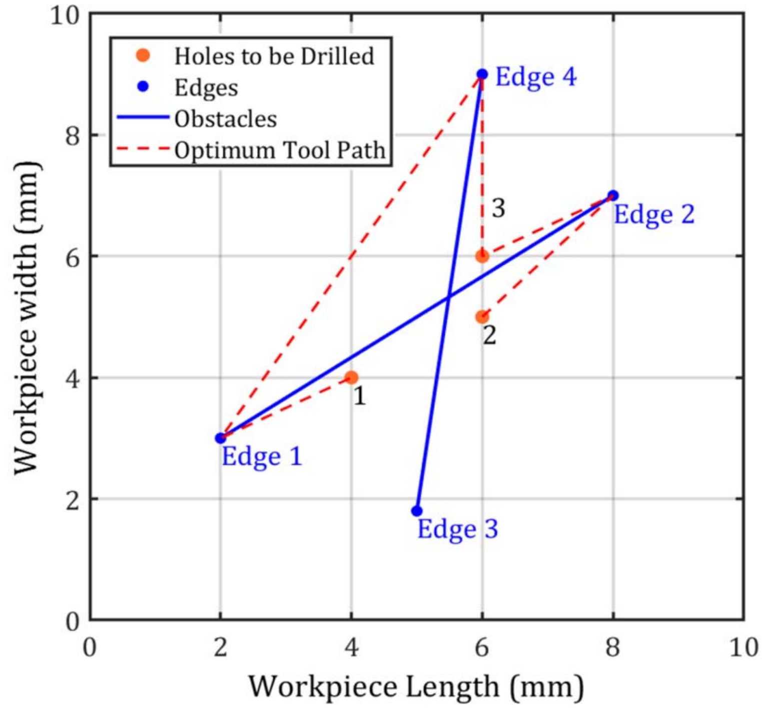


Figure 3.10: Near optimum path generated by second strategy for the case presented in Figure 3.7

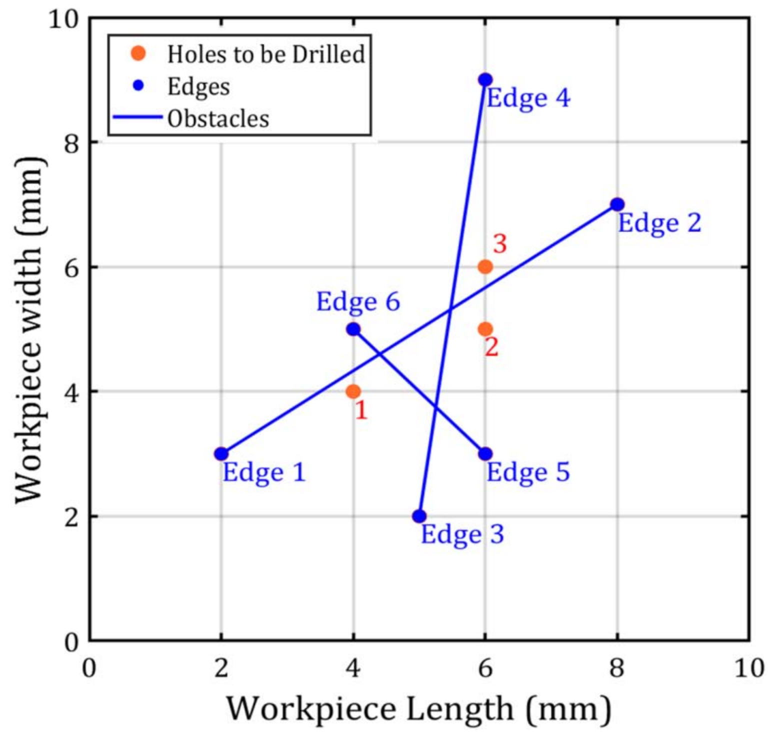


Figure 3.11: A complex scenario for which the second strategy is unable to deliver the near optimum tool path

3.6 Heuristic algorithm: General Local Search heuristic

In order to achieve a more robust algorithm capable of generating near optimum tool path for a wide range of simple to complex scenarios, the local search method will be implemented instead of the nearest neighborhood approach. Local search algorithms have been proposed during the mid-sixties to deal with computational difficulties of NP-hard problems and solve the TSP. Having an objective function f in a minimization/maximization problem and a feasible solution S, local search algorithm tries to construct and improve the feasible solutions in TSP [57]. In local search, once a current solution is achieved, the algorithms will explore to modify a better-quality solution within its neighbors/domains. The local search has the following steps:

- Generate an initial current solution S, and calculate the objective function
- Create new solution S' at every iteration and calculate the objective function
- Compare objective function of S and S'. If new solution S' is better than S, replace S with S′ and S′ becomes the new current solution
- Continue to reach the number of iterations

Iteration is a repetition which leads to move from one solution to another and varies case to case depending on the number of combinations. Local search generates new solutions in different ways. Creation of a new solution (step two) can be done by generating a new random tour like general local search or modifying some of its elements like k-opt algorithms in order to achieve a better solution [46, 51, 57-61]. Creating a completely new tour in some papers considered best improvement strategy [22, 34, 61].

3.6.1 Potential complexity#2 re-solved: workpiece with two intersecting wall obstacles

To find the near optimum solution for the example mentioned in Figure 3.7, a local search algorithm is applied. As previously shown in Table 3.5, the optimum tool path length varies depending on the tool origin. As a reason, a good selection of safe tool origin will minimize the total drilling path length and save time in mass production. In the field work, multiple workpieces are mounted on the CNC machine table, after one workpiece is being cut, the tool moves to the next workpiece and starts cutting, until all workpieces are being cut. There is no need for the drilling tool to return to the safe origin after drilling each workpiece. Hereafter, in this thesis, the tool starts from the safe tool origin and stops after the last hole being drilled to fulfill the mentioned situation. As presented in Figure 3.12, the local search algorithm is able to solve the example presented in Figure 3.7. The results are summarized in Table 3.7.

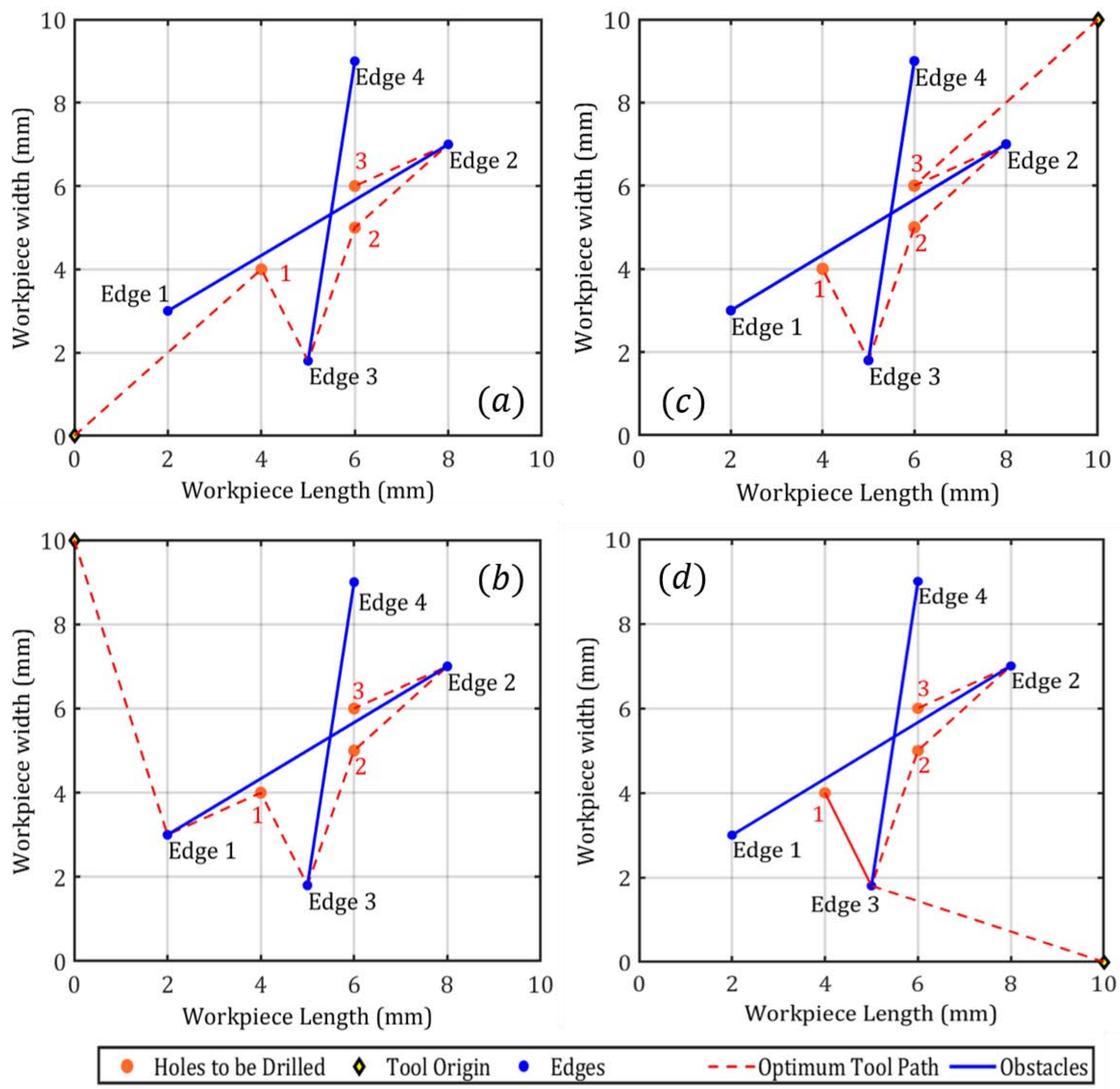


Figure 3.12: Near optimum tool path using proposed local search when tool origins is located at (a) point (0, 0), (b) point (0, 10), (c) point (10, 10), (d) point (10, 0), for the case presented in Figure 3.7

Table 3.7: Summary of near optimum generated tool paths for different tool origins (case in Figure 3.7)

Tool origin	**(0,0)**	**(0,10)**	**(10,10)**	**(10,0)**
Near optimum path length (mm)	16.49	20.35	16.49	18.56
Computational time (seconds)	4	4	4	5
Number of iterations	100	100	100	100
Path	$0 \rightarrow 1 \rightarrow 6 \rightarrow 2 \rightarrow 5 \rightarrow 3$	$0 \rightarrow 4 \rightarrow 1 \rightarrow 6 \rightarrow 2 \rightarrow 5 \rightarrow 3$	$0 \rightarrow 3 \rightarrow 5 \rightarrow 2 \rightarrow 6 \rightarrow 1$	$0 \rightarrow 6 \rightarrow 1 \rightarrow 6 \rightarrow 2 \rightarrow 5 \rightarrow 3$

Regarding the results, the path starts from the down left corner and top right corner shown in Figure 3.12 (a) and (c) respectively, has the minimum path length. Thus, the operator can define either two corners as the safe tool origin.100 is selected for the number of iterations, the selection criteria for number of iterations will be discussed in detail in the next chapter.

3.6.2 Potential complexity#3: workpiece with one circular and one straight obstacle

Another common feature in industrial workpieces is cylindrical geometry. In addition to straight obstacles, cylindrical obstacles may also be seen in machined parts. The approach to find the shortest path length with obstacles in the form of a circle is a bit different. If the obstacle is in the form of a circle, a tangent line has to be selected and then distance will be calculated. Detailed proof and mathematical calculations are available in Appendix A. In order to investigate the effectiveness of local search algorithms in presence of straight and cylindrical obstacles, an imaginary workpiece is selected.

Figure 3.13 shows the workpiece dimensions, locations of the holes to be drilled, and arrangement of the obstacles in the simulated scenario. In this scenario, the tool drills four holes on a workpiece with two obstacles in the form of a straight wall and a circle. Height of the cylindrical obstacle and the wall are 100 mm and 20 mm, respectively. The tool must detect the obstacles and move around them (2D) to avoid any collision. Table 3.8 shows the coordinates of each hole and the locations of the obstacles.

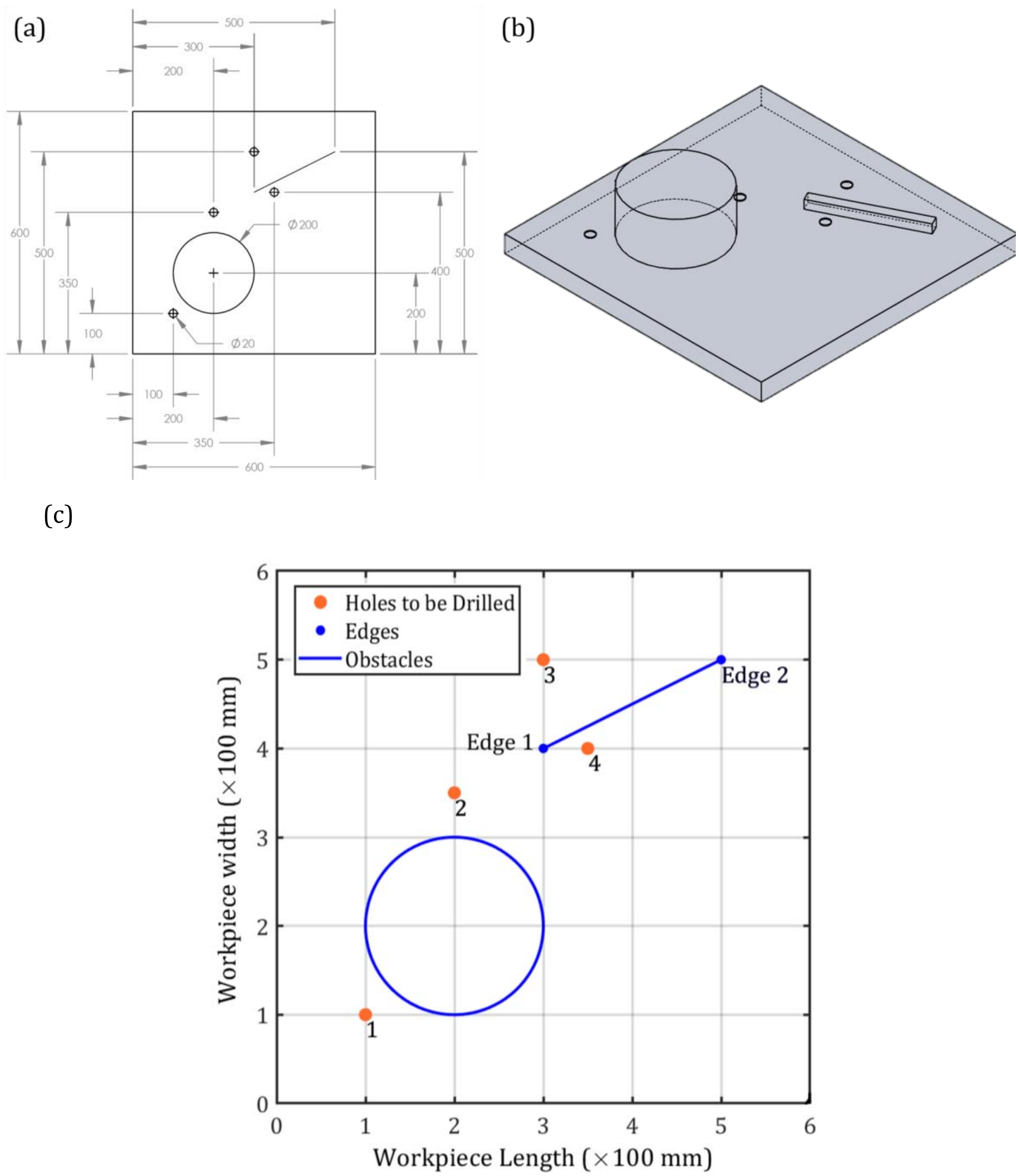


Figure 3.13: 4-hole drilling workpiece dimensions and arrangement of holes and obstacles (edges), (a) 2D drawing; (b) Isometric view (c) MATLAB figure (all dimensions are in mm)

Table 3.8: Location of holes and obstacles

No.	Hole coordinate (mm)				Straight obstacle coordinate (mm)		Circular obstacle center coordinate (mm)
	1	**2**	**3**	**4**	**1**	**2**	$r = 100$
x	100	200	300	350	300	500	200
y	100	350	500	400	400	500	200

The local search algorithm that is applied in this example is presented as follows.

Step 1. Initialize from the specified origin of the workpiece. Set it as the current city. Mark it as visited.

Step 2. Select a new random unvisited city from an array that includes both cities and edges' indexes. (Note that the edges indexes are placed after the cities indexes in the array). Set it as the next city in the path.

Step 3. Investigate whether the path from the current city to the next city has a collision with the obstacle.

3-1: If the path has a collision, then:

a: If the obstacle is in the form of a straight wall, go to step (2)

b: If the obstacle is in the form of a circle, draw and calculate the tangent line to the circular obstacle from the current city to the next city and go to step (4).

3-2: If the path has no collisions, go to step 4

Step 4. Set the next city as the current city in the tour. Mark this city as visited.

Step 5. Run the algorithm until all cities are visited.

If all cities in the domain are visited, then terminate the loop and go to step 6.

Else go to step 2 (This will form one tour as a group of all cities to be visited).

Step 6. Calculate the overall travelling distance for all of the tours (group of all cities that has been previously created including the origin). Select the tour with the minimum overall distance among all iterations.

The flowchart for the local search algorithm is presented in Figure 3.14.

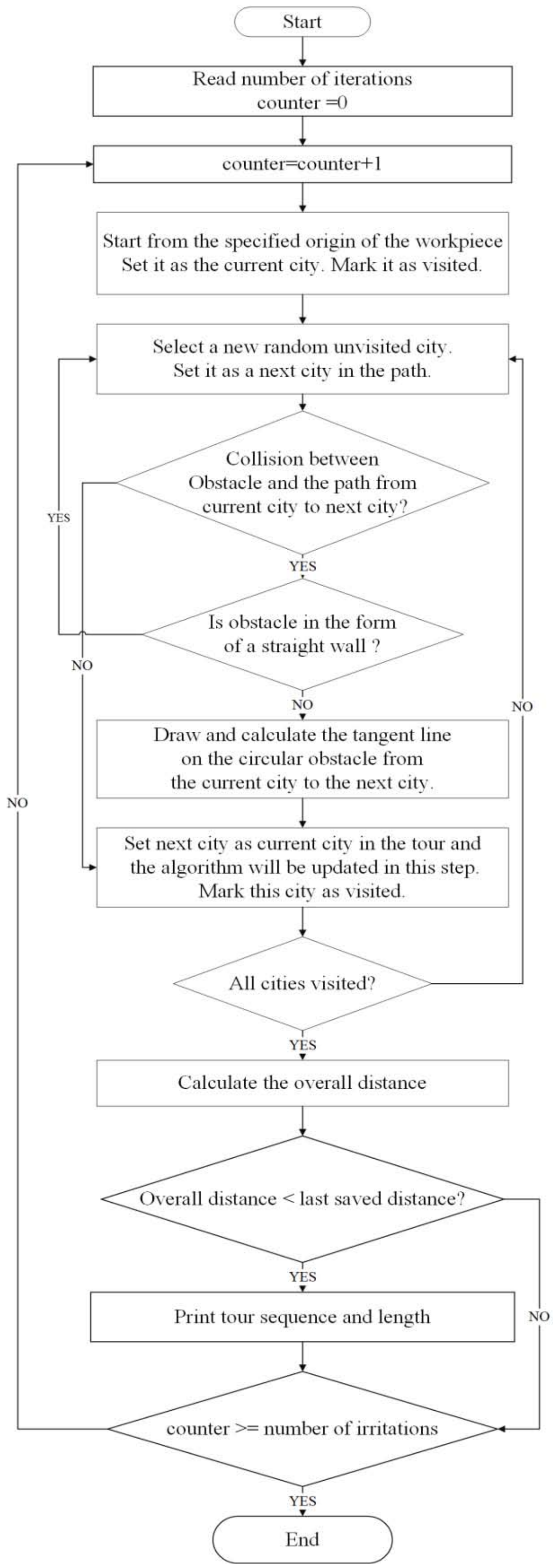


Figure 3.14: Proposed local search flowchart

For the problem presented in Figure 3.13, the algorithm was executed for four different scenarios where the safe tool origin is located at (0, 0), (0, 600), (600, 600), and (600, 0) (see Figure 3.15).

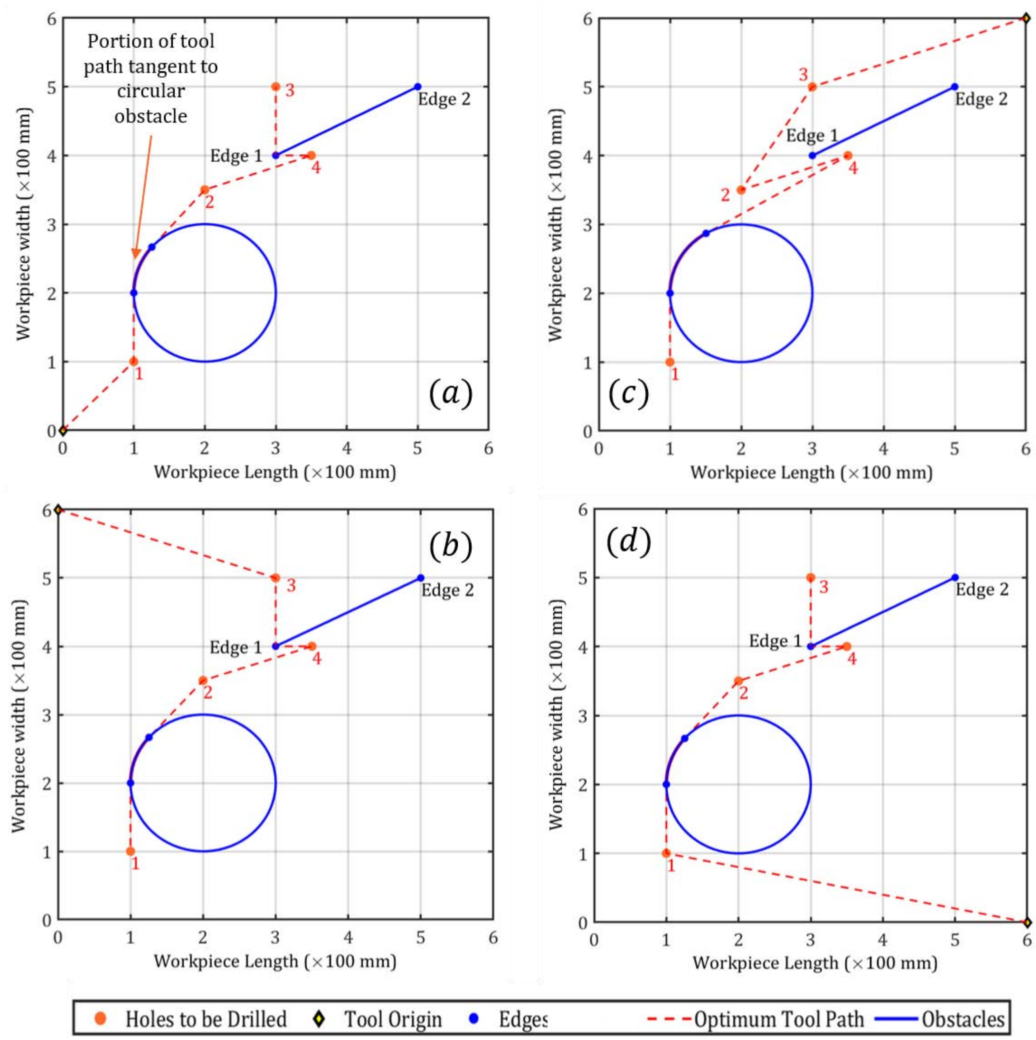


Figure 3.15: Near optimum tool path when tool origins is located at (a) point (0, 0), (b) point (0, 600), (c) point (600, 600), (d) point (600, 0) for the case presented in Figure 3.13

As presented in Figure 3.15, the near optimum TSP tour when the starting point of motion is located at (0, 0) is : the tool origin (0, 0), hole 1 (e.g. city 1), portion of the circular obstacle, hole 2, hole 4, edge 1, and hole 3. The near optimum path length is 734 mm. The total run time is 38.24 seconds for 200 iterations. The algorithm can be executed for the remaining corners as shown in Figure 3.15 (b), (c), (d).

Again, tool origin selection affects the overall tool path. The results are summarized in Table 3.9. For ease of tracing the optimum tool path, edge 1 is indexed as 5, edge 2 is indexed as 6, and tool origin is indexed as 0. Needless to mention that due to the nature of local search algorithm, the optimum tool paths may be local optimums. Global optimum for each case can be obtained by inspecting all possible combinations which is extremely time consuming; specially, when the number of cities and obstacles increase.

Table 3.9 : Summary of near optimum generated tool paths for different tool origins (case in Figure 3.13)

Tool origin	**(0,0)**	**(0,600)**	**(600,600)**	**(600, 0)**
Near Optimum path length (mm)	734	909	1089	1103
Computational time (seconds)	38.24	28.83	34.88	28.09
Number of iterations	200	200	200	200
Path	0 → 1 → circle → 2 → 4 → 5 → 3	0 → 3 → 5 → 4−→ 2 → circle → 1	0 → 3 → 2 → 4 → circle → 1	0 → 1 → circle → 2 → 4 → 5 → 3

Regarding the results, the path starts from the down left corner has the minimum path length. Thus, (0, 0) can define as the safe tool origin by the operator.

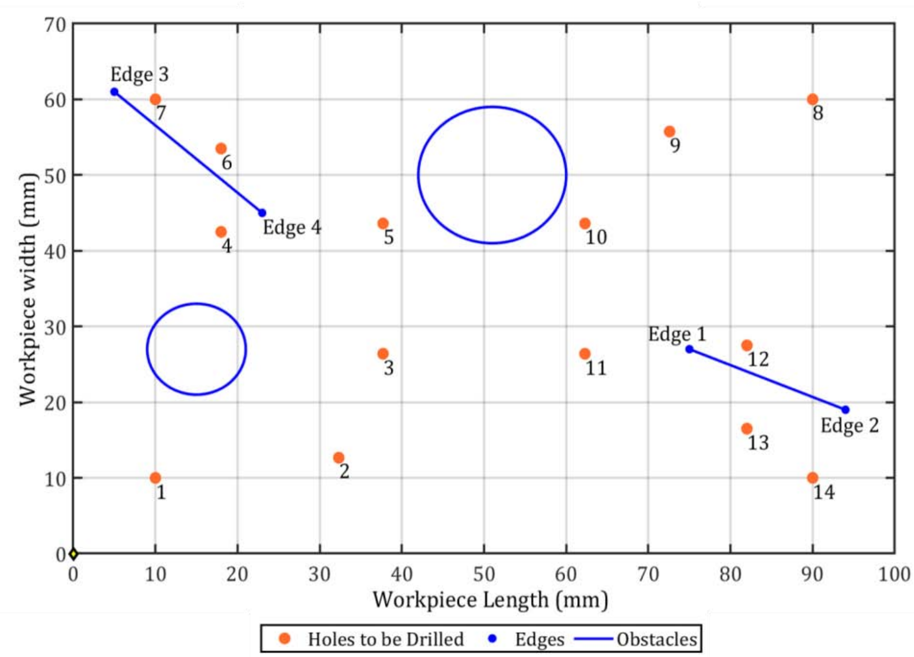
3.6.3 Potential complexity#4: workpiece with circular and straight obstacles

In this scenario, 14-hole drilling workpiece shown in Figure 3.4 is used. In this case though, two obstacles in the form of a straight wall and two circle obstacles are added to the problem in Figure 3.4. Height of the cylindrical obstacles are 15 and 25 mm and height of walls are 10 and 20 mm. Figure 3.16 shows the workpiece dimensions, locations of the holes to be drilled, and arrangement of the obstacles in this scenario. Table 3.10 shows the locations of the obstacles.

Table 3.10: Location of obstacles

No.	Straight obstacle coordinate (mm)				Circular obstacle center coordinate (mm)	
	1	2	3	4	$r = 6$	$r = 9$
x	75	94	5	23	15	51
y	27	19	61	45	27	50

(a)



(b)

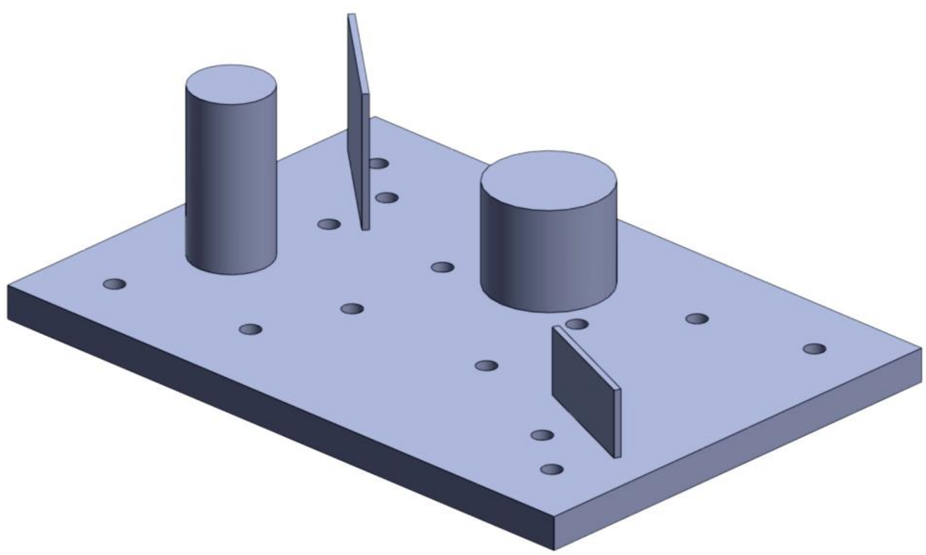


Figure 3.16: 14-hole drilling workpiece dimensions and arrangement of holes and obstacles, (a) MATLAB figure (b) Isometric view (all dimensions are in mm)

Four different tool origins, located at (0, 0), (0, 70), (100, 70), and (100, 0) for the problem presented in Figure 3.16, was executed (see Figure 3.17). The results are summarized in Table 3.11.

Regarding the results, the path starts from the bottom left corner shown in Figure 3.17 (a) has the minimum path length. Thus, the bottom left corner can be defined as the safe tool origin.

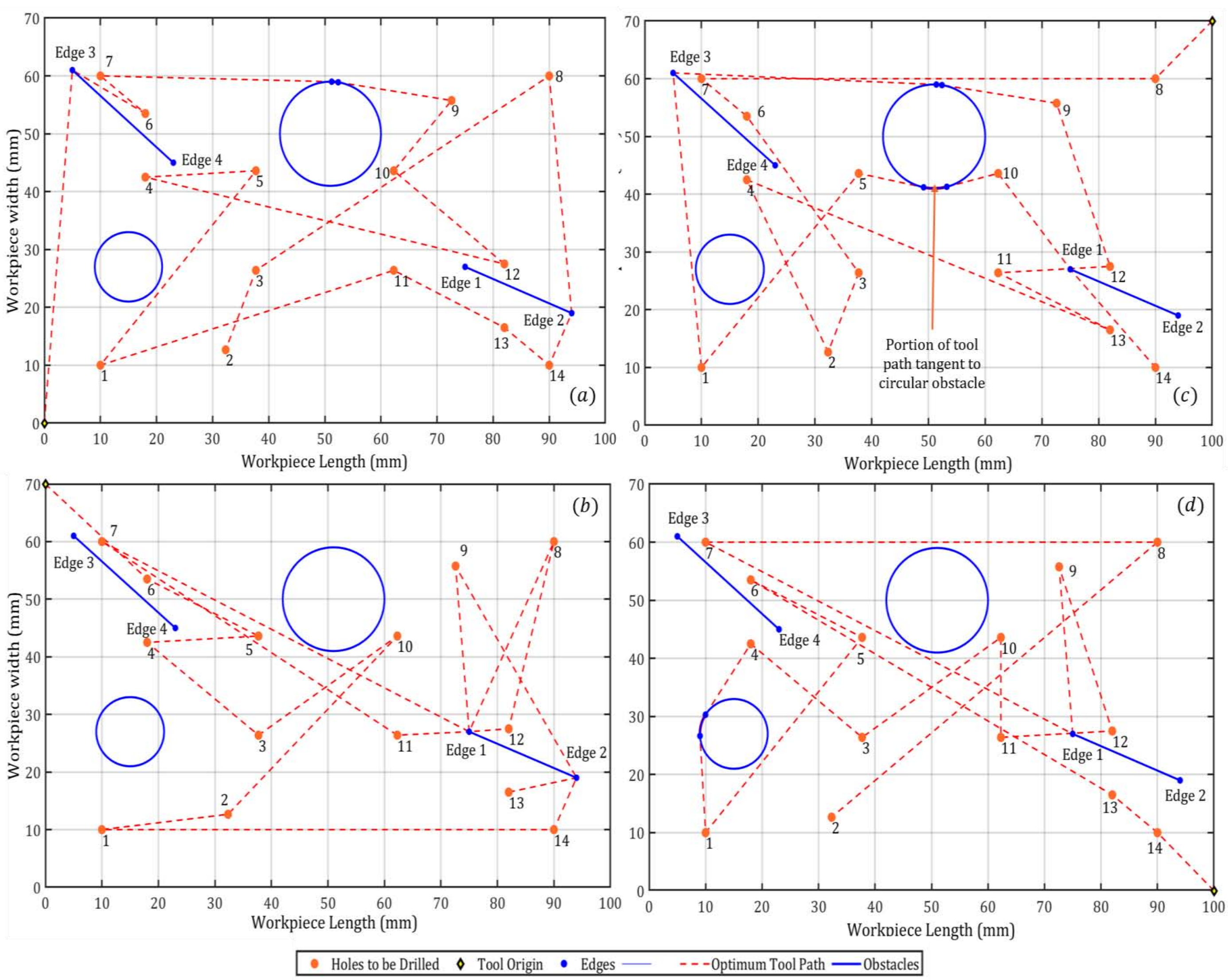


Figure 3.17: Near optimum tool path when tool origins is located at (a) point (0, 0), (b) point (0, 70), (c) point (100, 70), (d) point (100, 0) for the case presented in Figure 3.16

Table 3.11: Summary of near optimum generated tool paths for different tool origins (case in Figure 3.16)

Tool origin	**(0,0)**	**(0,70)**	**(100,70)**	**(100, 0)**
Near Optimum path length (mm)	535	606	558	578
Computational time (seconds)	1050	983	1063	992
Number of iterations	3000	3000	3000	3000
Path	$0 \to 17 \to 6 \to 7 \to$ circle $\to 9 \to 10 \to 12 \to 4 \to 5 \to 1 \to 11 \to 13 \to 14 \to 16 \to 8 \to 3 \to 2$	$0 \to 6 \to 5 \to 4 \to 3 \to 10 \to 2 \to 1 \to 14 \to 16 \to 9 \to 15 \to 12 \to 8 \to 15 \to 7 \to 11 \to 15 \to 16 \to 13$	$0 \to 8 \to 7 \to 6 \to 3 \to 2 \to 4 \to 13 \to 11 \to 12 \to 9 \to$ circle $\to 17 \to 1 \to 5 \to$ circle $\to 10 \to 15 \to 14$	$0 \to 14 \to 13 \to 6 \to 5 \to 1 \to$ circle $\to 4 \to 3 \to 10 \to 11 \to 12 \to 9 \to 15 \to 7 \to 8 \to 2$

3.7 Summery

The results presented in this section prove that the proposed model is able to achieve the shortest tool path length when drilling multiple holes on a workpiece. This is while the most common types of obstacles in practical applications, namely straight and circular profiles, are considered by the model. Also, the developed model considers the safe tool origin and optimizes the path accordingly to achieve the shortest path length.

Chapter 4: Model validation and results

4.1 Preamble

The main objective of this thesis is generating a collision free airtime tool path optimization in drilling. The steps that have been incorporated include modification of the TSP problem, investigation of the effects of tool origin, customizing the algorithm to collision free constraints, and finally implementing different scenarios. This chapter contains discussions regarding validation step, comparison step along with results of the presented works, stopping criteria, a brief description of the main contributions, and outline of the road map for future works. In this chapter Autodesk HSMWorks CAM software add-in to SOLIDWORKS is used for modelling and G-code generation. The G-code simulation is constructed by the Autodesk HSM Editor.

4.2 Complexity added to TSP by adding more elements

TSP is a class of NP-hard problems whose time complexity is exponential. To describe more, the solution for the TSP problem lies in the possibility of finding the best/possible solution within a great number of possible combinations. The number of possible combinations in a symmetric TSP problem is presented in equation 3.14. If we increase the number of cities or a few numbers of elements, the number of possible combinations quickly gets out of hand. Adding the collision free element in the TSP problem, increases the complexity even more compared to a common hole drilling problem. Modifying equation 3.14 to satisfy the collision free elements of the proposed algorithm:

$$Possible\ combinations\ in\ the\ proposed\ algorithm = ((number\ of\ holes + number\ of\ edges) - 1)!/2 \quad 4.1$$

According to equation 3.14, possible combinations of a four-hole problem with no obstacle is $\frac{(4-1)!}{2} = 3$, while possible combinations of the same problem with only one straight obstacle will significantly increase to $\frac{((4+2)-1)!}{2} = 60$ (see equation 4.1 and Figure 4.1).

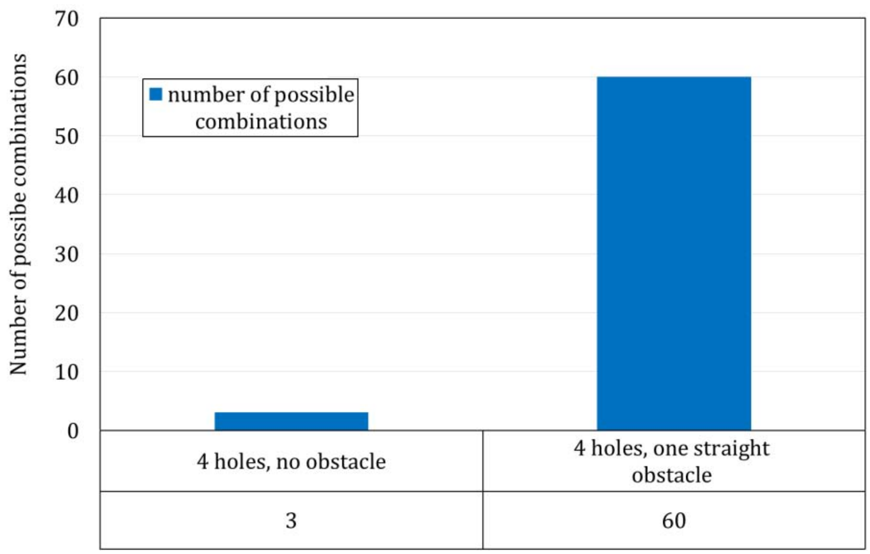


Figure 4.1: Complexity added by adding more elements

Adding circular obstacles to proposed algorithm even strikingly increases the possible combinations/complexity of the problem. Considering the example above adding one circle to the example, the possible combinations would be 181,440 according to equation 4.2.

$$Possible\ combinations\ in\ the\ proposed\ algorithm = ((number\ of\ holes + number\ of\ edges + number\ of\ circles * 4) - 1)!/2 \quad 4.2$$

Table 4.1: Complexity added to a four- hole problem by adding a straight and a circular obstacle

Number of holes and elements	4 holes, no obstacle	4 holes, one straight obstacle	4 holes, one straight obstacle, one circular obstacle
Number of possible combinations	3	60	181,440

4.3 Validation step

The proposed algorithm has been verified using an example shown in Figure 3.4, a case study applied in [36, 54-56]. To thoroughly check the performance of the proposed algorithm, the results are compared with the findings of the above works. In addition, the same example was modeled in CAD software and tool path was generated by HSMWorks software to check the performance of the optimization approaches over an industrial CAM software. The results are presented below in Table 4.2.

Table 4.2: Comparison of near optimum tool path generated by the proposed algorithm with [36, 54-56] and HSMWorks software

	HSMWorks	Proposed algorithm	Zhu [54]	Zhu and Zhang [55]	Kentli and Alkaya [36]	Aziz, et al. [56]
Algorithms	NN (based on [18])	Local Search	PSO	PSO	Modified Local Search	ssSKF
Optimum/Near optimum tool path (mm)	382	291	Best 280 Worst 307	Best 280 Worts 295	290	280

single-solution Simulated Kalman Filter (ssSKF)

In Table 4.2, in HSMWorks output, retraction level (automatically) defined as $Z = 0.1$, i.e. the height that the tool moves up to before the next cutting pass, with no modification to G-code. Hajad, et al. [18] mentioned that the suggested tool path in CAD/CAM software is generated based on the nearest neighborhood heuristic algorithm. In the work of Zhu and Zhang [55] only tool paths with less than 295 mm are listed. In all the mentioned works, the iteration number has not been stated as well as computational time. The proposed local search ran for 665 seconds with 10000 iterations.

As can be concluded from Table 4.2, the proposed local search is able to generate a tool path with a close convergence and accuracy to the results of mentioned works [36, 54-56] in a reasonable amount of time. The results from Table 4.2 emphasize the fact that the tool path generated by CAM software is not optimal, and almost 36% higher than the best-known tool path in a 14-hole workpiece. This percentage becomes more significant with more complex workpieces, in mass production the extra time is consumed for each single workpiece, hence any reduction in tool path can save a lot of time in mass production of complex parts.

4.4 Comparison step

In everyday machining practice CAD/CAM is usually utilized to design the part and generate the corresponding tool path for subsequent machining processes. In such a routine process, the part is initially designed by CAD software and the solid model is then imported to CAM software for post-processing, creating tool path, and ultimately generating G-code for the CNC machine. The post processor generates the tool path such that any unwanted collision between the cutting tool and workpiece stock is avoided. This is typically achieved by selecting the stock top, i.e. highest silhouette of the workpiece, plus a predefined offset, i.e. clearance height, as retraction height. Thus, in 3-axis machining, the cutting tool usually moves up to clear obstacles and reach the desired destination. In such a strategy, the generated tool path is not necessarily the optimum path.

This may not be considered an issue for a single job; however, it results in significant loss of time and revenue in high quantity batch production. Therefore, finding the optimum or near optimum tool path with the least travel distance is very advantageous. Considering the example shown in Figure 3.13, the results obtained from the developed algorithm with the automatically produced tool paths using HSMWorks CAM software are compared in Figure 4.2 to Figure 4.5.

The total tool path length in each figure is specified in the red box. Note that G-code for the near optimum tool path generated by the proposed algorithm was manually written and fed to Autodesk HSM Editor in order to visualize and compare the results. For all simulations, no offset or clearance height is selected. Feed height (the height to which the tool moves rapidly before changing the feed rate to enter the part and start cutting) is also selected as zero. Since the focus of the present thesis is generating near optimum collision-free tool path and not the mechanics of drilling, zero depth was assumed for the holes in the proposed algorithm and CAD/CAM simulation. The objective is to travel between the holes in an optimized manner. Depth of holes to be drilled will definitely affect the machining time; nevertheless, it will be the same between the two approaches and therefore will not affect the comparison.

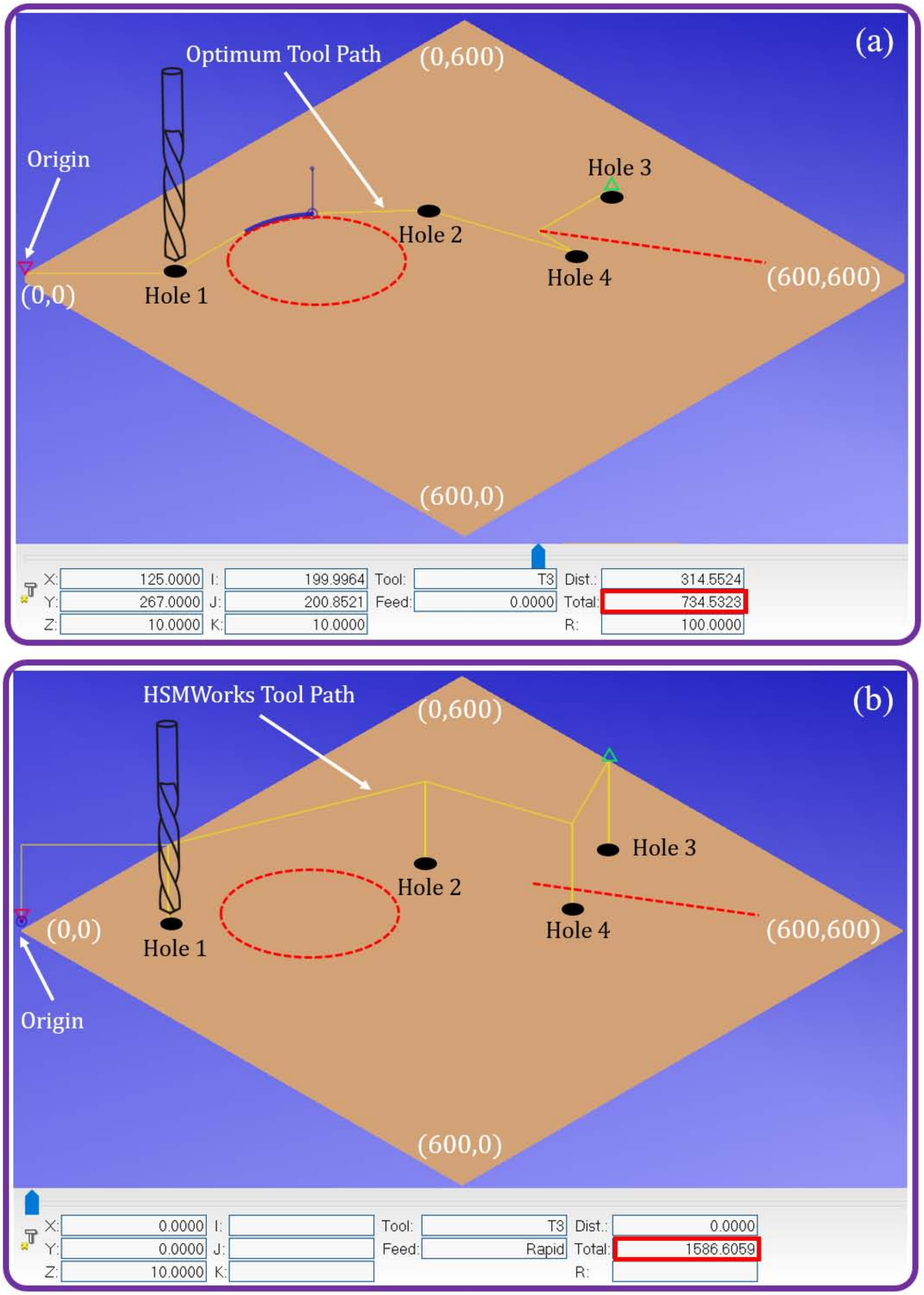


Figure 4.2: Comparison of the tool path length when tool origin is located at (0,0), (a) near optimum path generated by the proposed algorithm, (b) the automatically generated path by HSMWorks

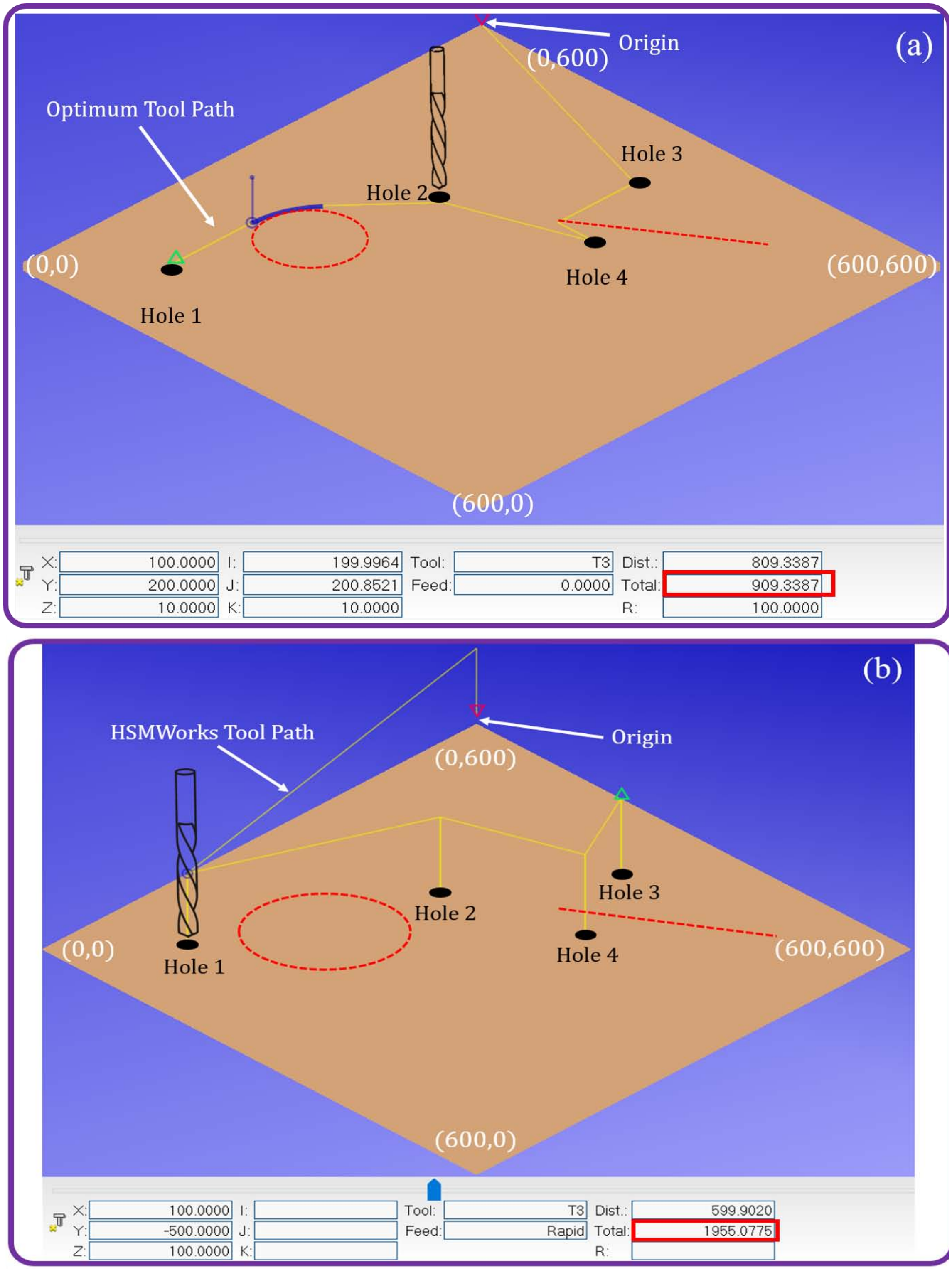


Figure 4.3: Comparison of the tool path length when tool origin is located at (0,600), (a) near optimum path generated by the proposed algorithm, (b) the automatically generated path by HSMWorks

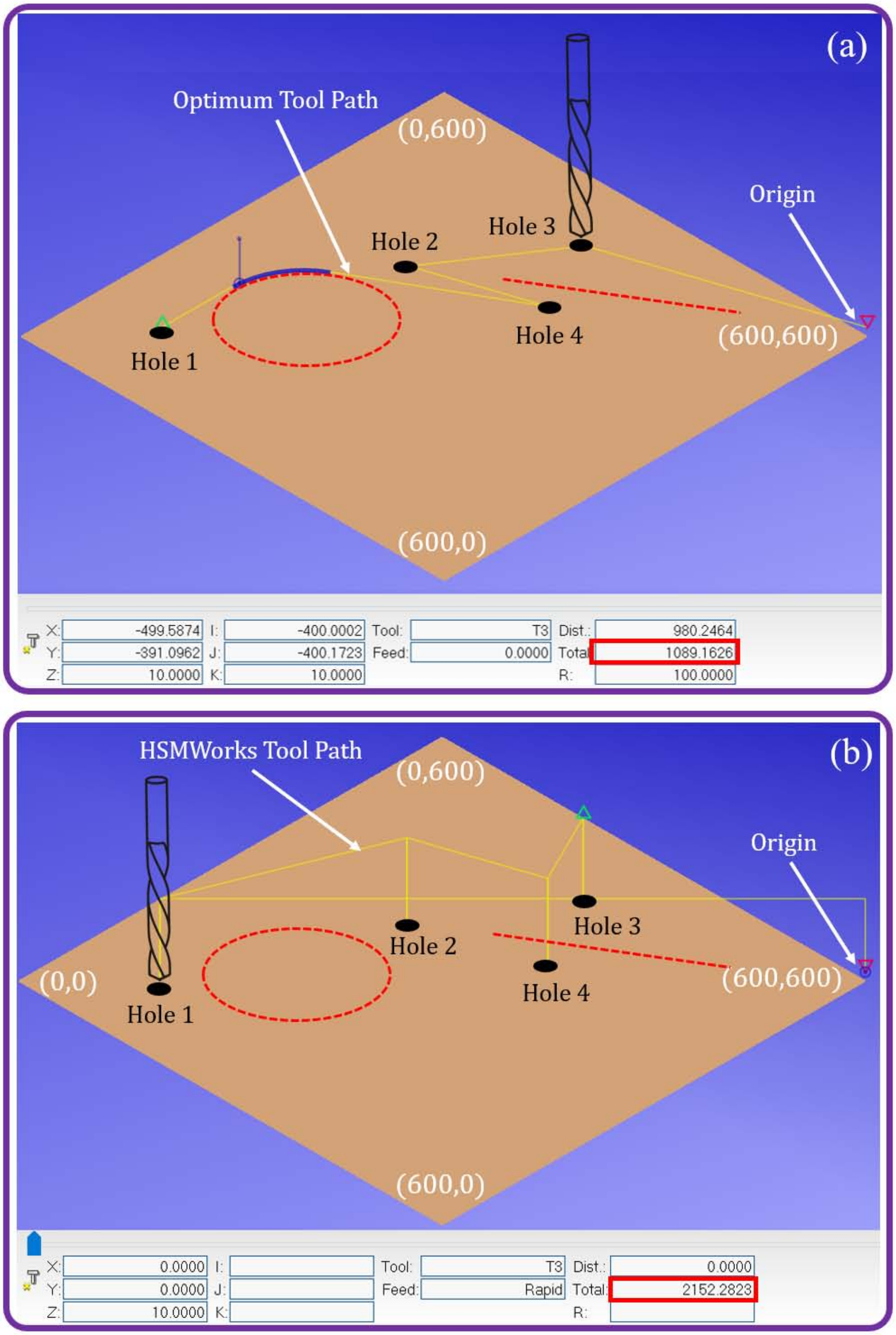


Figure 4.4: Comparison of the tool path length when tool origin is located at (600,600), (a) near optimum path generated by the proposed algorithm, (b) the automatically generated path by HSMWorks

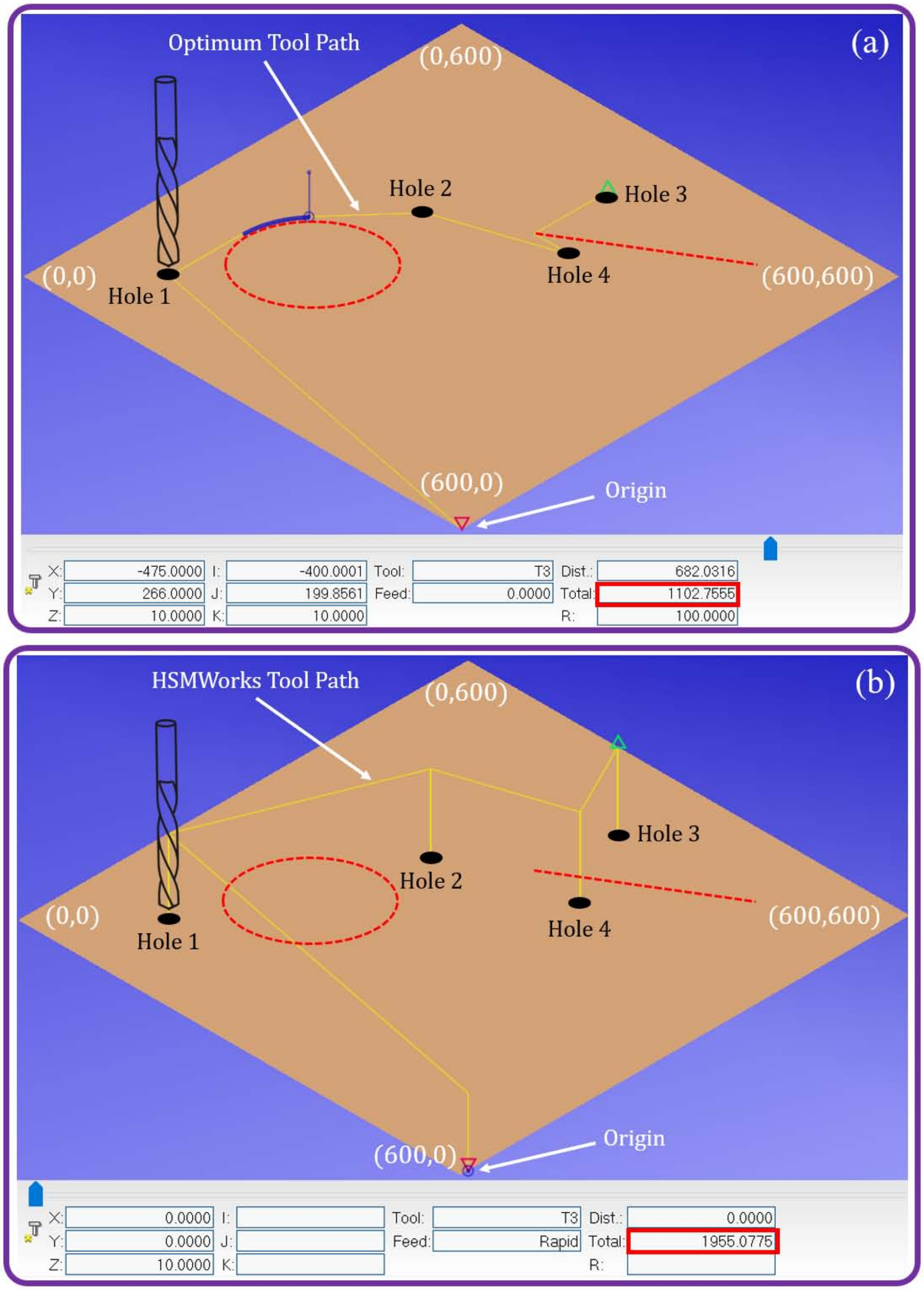


Figure 4.5: Comparison of the tool path length when tool origin is located at (600,0), (a) near optimum path generated by the proposed algorithm, (b) the automatically generated path by HSMWorks

Table 4.3 summarizes the results of comparison between the length of near optimum tool path generated by the proposed algorithm and the automatically generated tool path by HSMWorks.

Table 4.3: Comparison of proposed algorithm results with HSMWorks CAM software

Tool origin	**(0,0)**	**(0,600)**	**(600,600)**	**(600,0)**
Near Optimum path length (mm)	734	909	1089	1103
HSMWorks tool path length (mm)	1586.60	1955.08	2152.28	1955.08

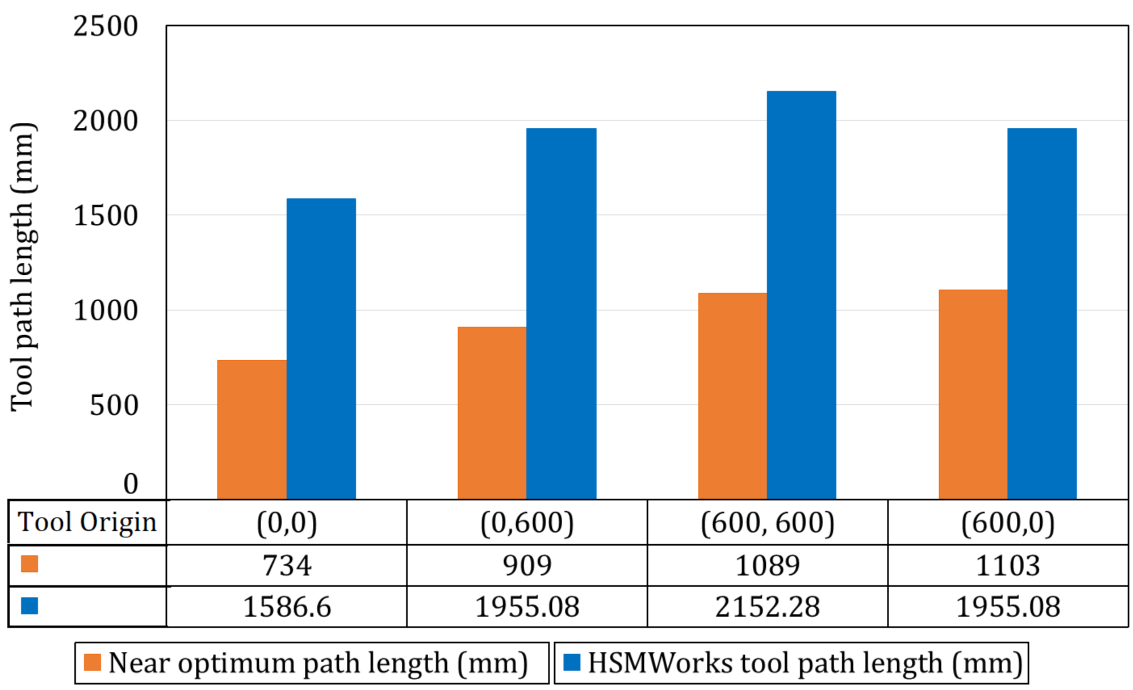


Figure 4.6: Comparison of the tool path length when tool origin is located at (0,0), (0,600), (600, 600) and (600,0) for the case presented in Figure 3.13

As can be seen in Figure 4.6, length of the path generated by the proposed algorithm is considerably shorter than the software-generated ones in all cases. In all cases the tool path generated by the proposed algorithm is more than 50% shorter than the path generated by

HSMWorks CAM. The results also found to be proportional, the higher the tool retraction height, the higher the improvement in the reduction of the total tool path length. Additionally, larger the number of holes, higher the improvement in tool path is seen.

In the HSMWorks CAM, the cutting tool moves upward (in *z* direction) to reach the clearance height and then moves through the space above the workpiece stock to reach the next destination. Since the height of the largest feature (e.g. obstacle) on this part, which is the circular obstacle, is 100 mm, the retraction height is automatically set to 100 mm for the workpiece stock. That means, to prevent collision between the tool and obstacles in the aforementioned example, the tool has to move upward 100 mm to clear the obstacle with the largest height. The tool will then need to move down 100 mm to drill the next hole. Note that although the height of the straight obstacle (wall shape feature) is 20 mm, the CAM software still considers the retraction height as 100 mm, which means the feature with largest height determines the retraction height for the entire workpiece stock.

Consequently, the tool path becomes significantly larger than the near optimum path generated by the proposed algorithm. Needless to mention that although the example presented is quite simple with only four holes and two obstacles, the difference between the near optimum tool path and the automatically generated one by the CAM software is noteworthy. Thus, this difference for parts with more complex geometry will definitely be more significant. Furthermore, changing the tool origin has no effect on the visiting sequence of holes in the path automatically generated by the CAM software. However, in the proposed model, the sequence changes with the tool origin to deliver the near optimum tool path.

For the example shown in Figure 3.16, the results obtained from the proposed algorithm and the HSMWorks are compared (see Figure 4.7 to Figure 4.10).

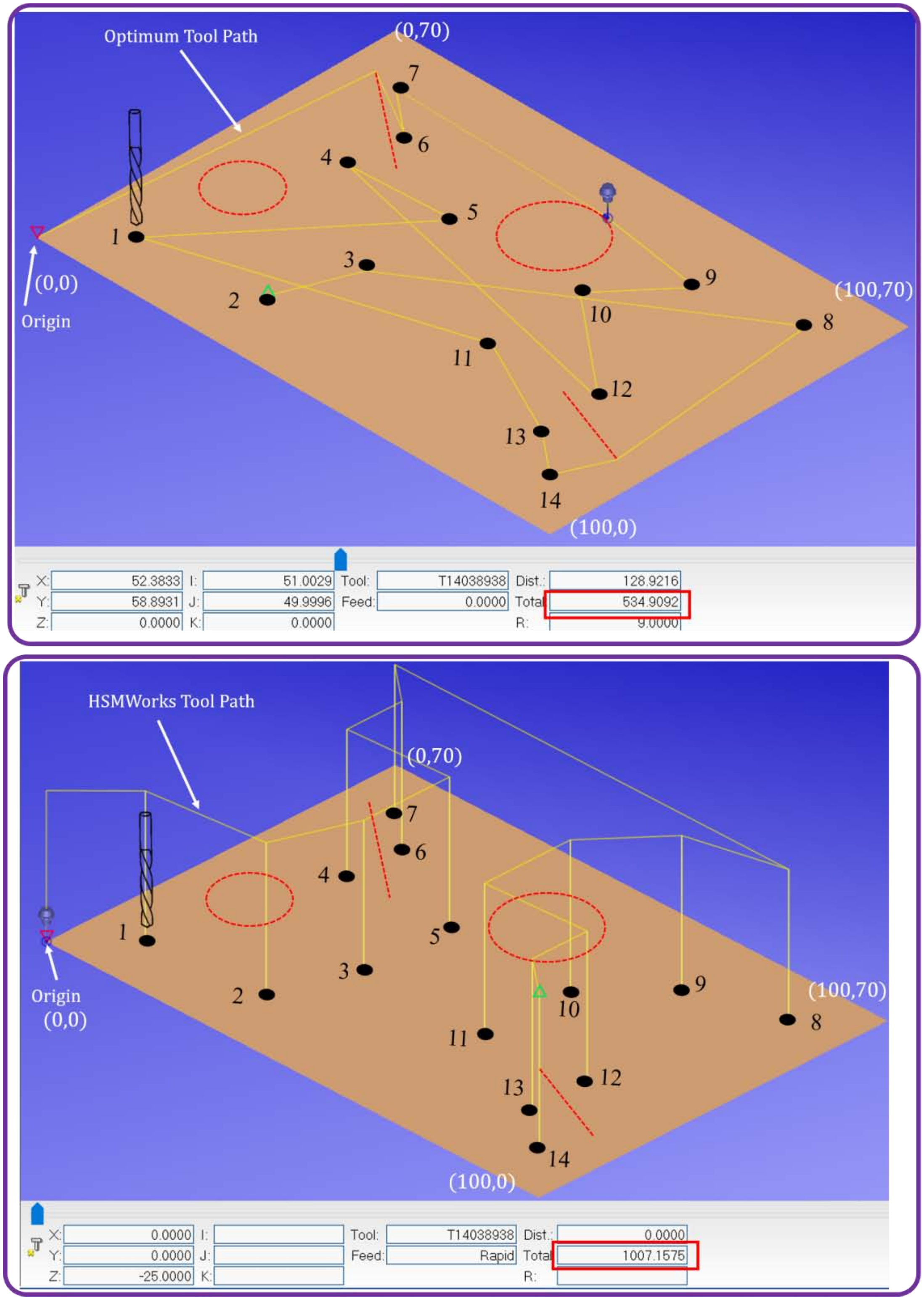
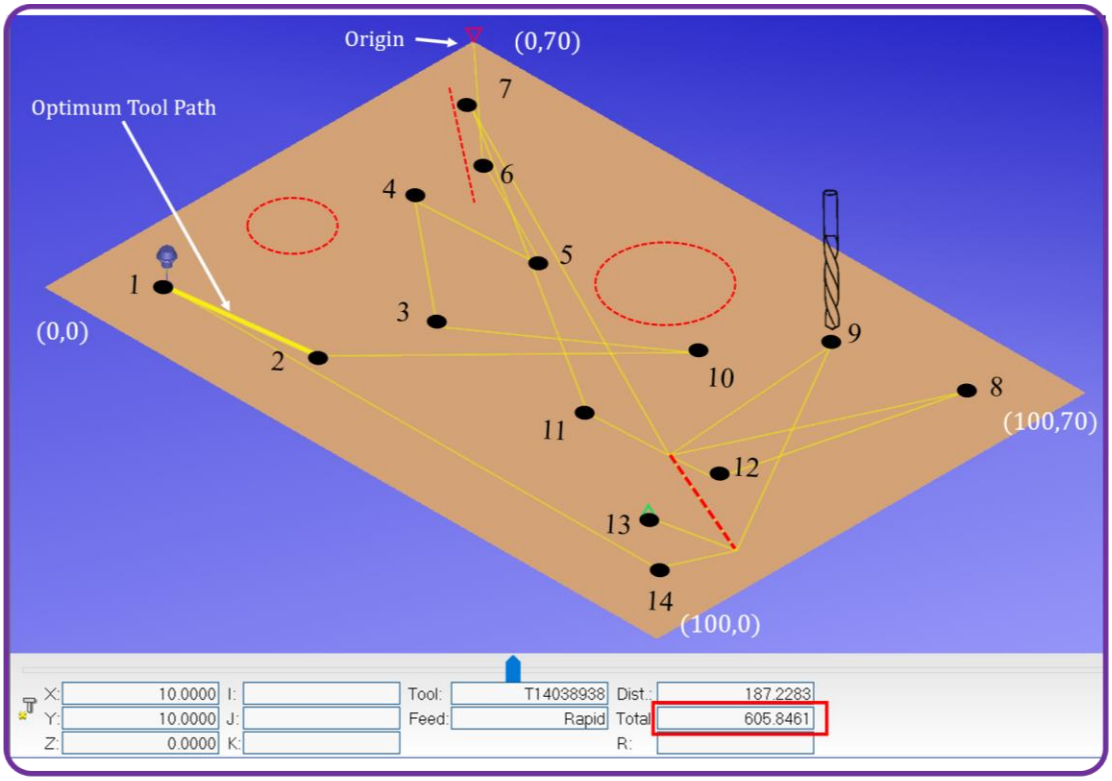


Figure 4.7: Comparison of the tool path length when tool origin is located at (0,0), (a) near optimum path generated by the proposed algorithm, (b) the automatically generated path by HSMWorks



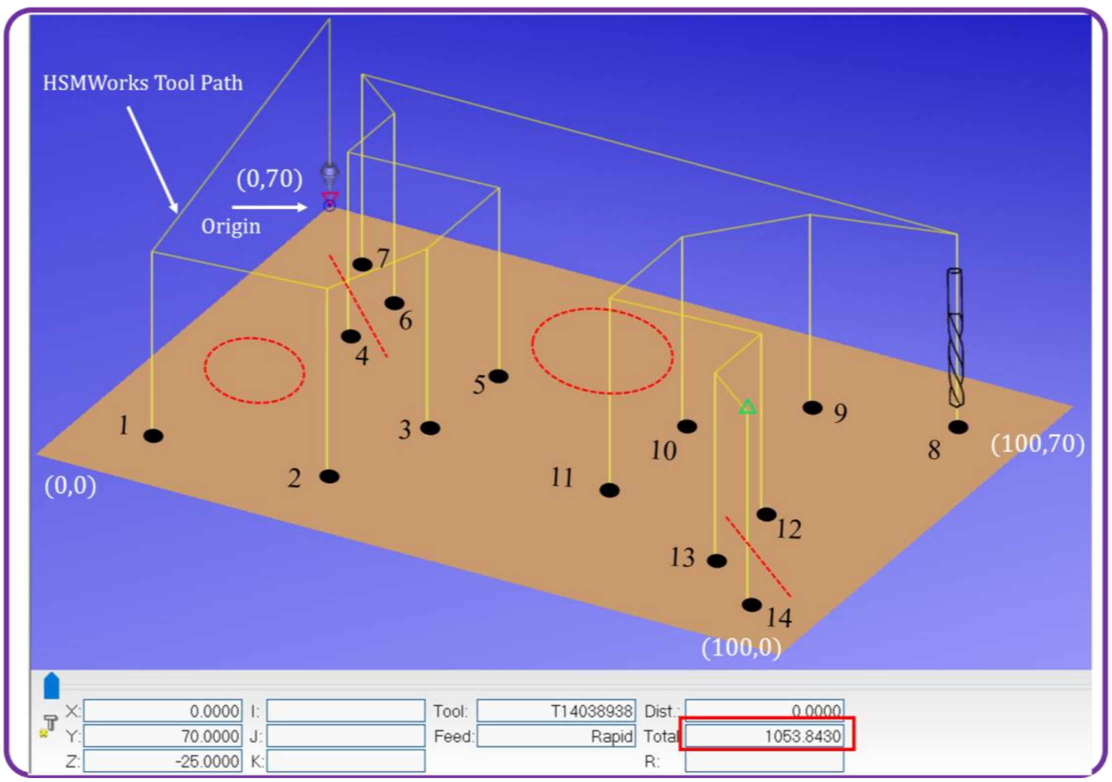


Figure 4.8: Comparison of the tool path length when tool origin is located at (0,70), (a) near optimum path generated by the proposed algorithm, (b) the automatically generated path by HSMWorks

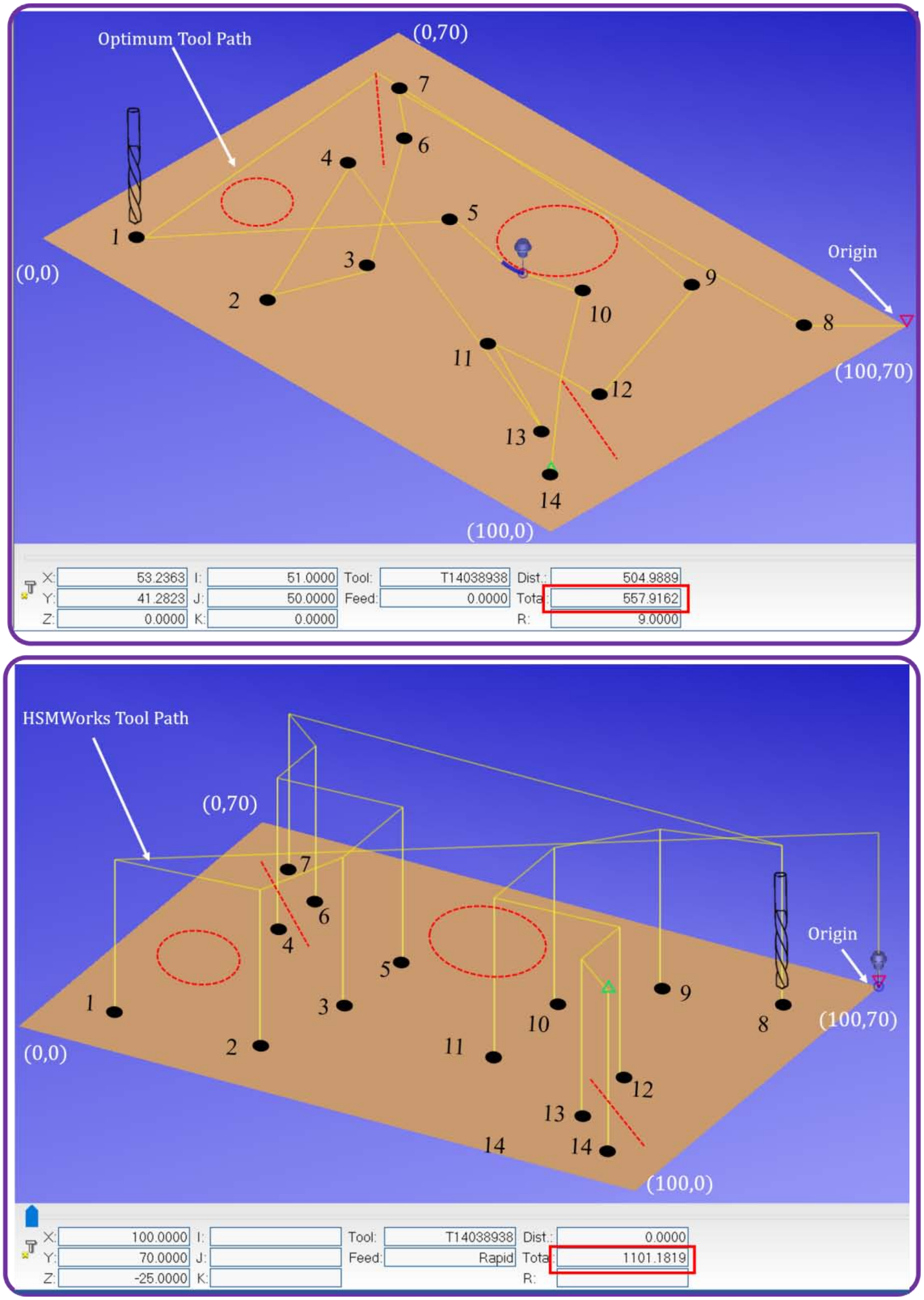


Figure 4.9: Comparison of the tool path length when tool origin is located at (100,70), (a) near optimum path generated by the proposed algorithm, (b) the automatically generated path by HSMWorks

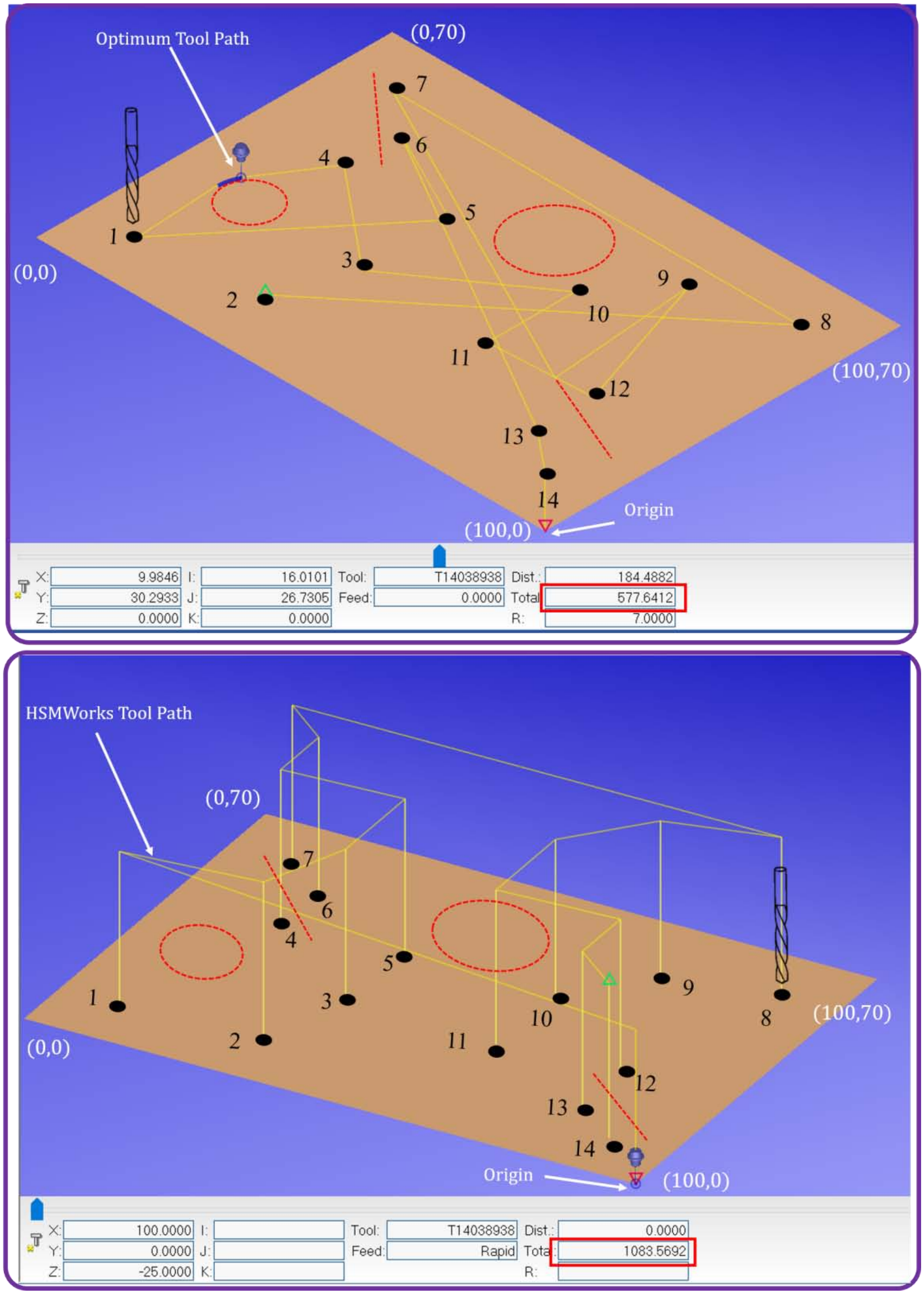


Figure 4.10: Comparison of the tool path length when tool origin is located at (100,0), (a) near optimum path generated by the proposed algorithm, (b) the automatically generated path by HSMWorks

Table 4.4 summarizes the results of comparison between the length of near optimum tool path generated by the proposed algorithm and the automatically generated tool path by HSMWorks.

Table 4.4: Comparison of proposed algorithm results with HSMWorks CAM software

Tool origin	**(0,0)**	**(0,70)**	**(100,70)**	**(100,0)**
Near Optimum path length (mm)	535	606	558	578
HSMWorks tool path length (mm)	1007	1054	1101	1048

Similar to the previous example, length of the path generated by the proposed algorithm is considerably shorter than the software-generated ones in all corners. Tool path generated by the proposed algorithm is more than 50% shorter than the path generated by CAD/CAM. Consequently, the higher the feature height or the larger the number of holes, the improvement in the reduction of the total tool path length is much significant (see Figure 4.11).

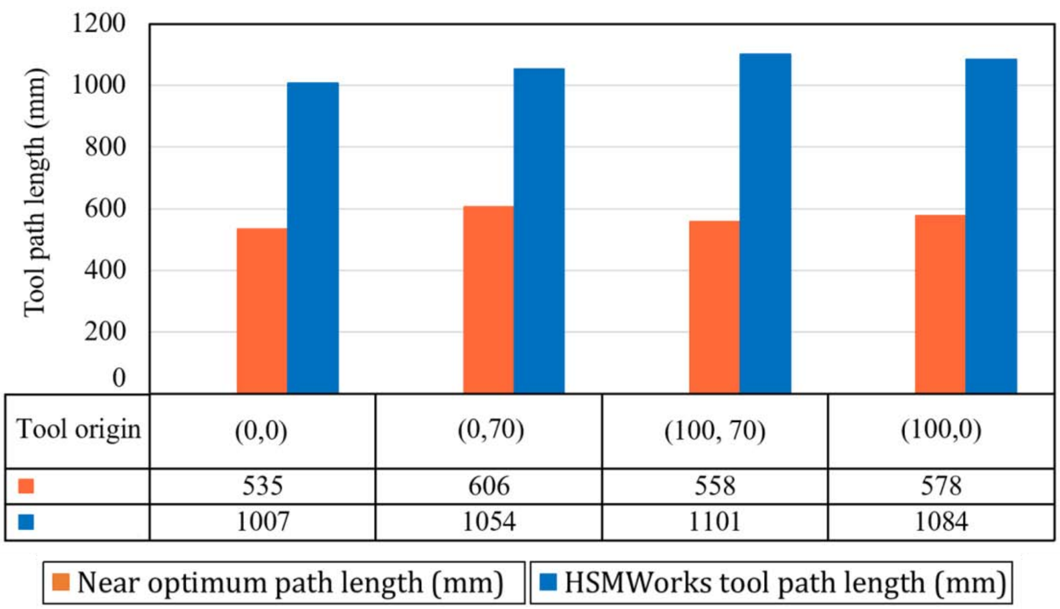


Figure 4.11: Comparison of the tool path length when tool origin is located at (0,0), (0,70), (100, 70) and (100,0) for the case presented in Figure 3.16

In the CAM software, the cutting tool clears the obstacles by moving upward (in *z* direction) to reach the retraction height. The highest feature (e.g. obstacle) on this part, circular obstacle, is 25 mm, since retraction height is automatically set to 25 mm. The tool has to move upward 25 mm to clear the obstacle with the largest height and then need to move down 25 mm to drill the next hole. Height of the other obstacles (wall shape features and the other circular shape feature) are 10, 15, 20 mm, the CAM software still considers the retraction height as 25 mm. Again, the highest feature defines the retraction height for the entire workpiece stock.

Consequently, the higher the feature height, the tool travel distance is much larger compared to near optimum path generated by the proposed algorithm. In addition, in the proposed model, the effects of any sequence changes in tool origin to deliver the near optimum tool path, is investigated.

4.5 Stopping Criteria

Factors like larger problem size, using metaheuristics over heuristics and selection of the stopping criterion will cause computational time to increase. In industry especially manufacturing, time is an important factor. The idea of the whole optimization problem is finally decreasing the manufacturing time in order to survive in the fast pace competition world. Any fraction of reduction of time in machining processes matters a lot. Stopping criterion can select according to the judgment of the user and often determined by the time and level of optimality.

The near optimum solution for example mentioned in Figure 3.16 is generated when drill bit stars from bottom left corner. Figure 4.12 demonstrates the near optimum tool path generated in each iteration until reaching the maximum number of iterations which is 3000.

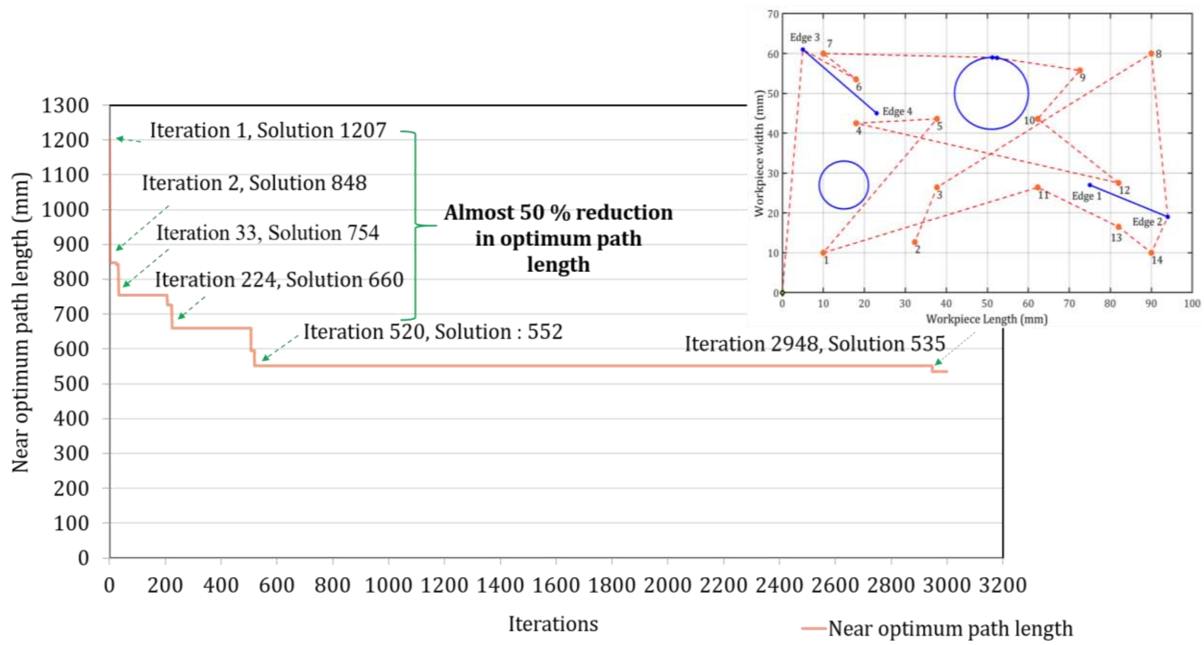


Figure 4.12: Near optimum path length in each iteration for the case presented in Figure 3.16

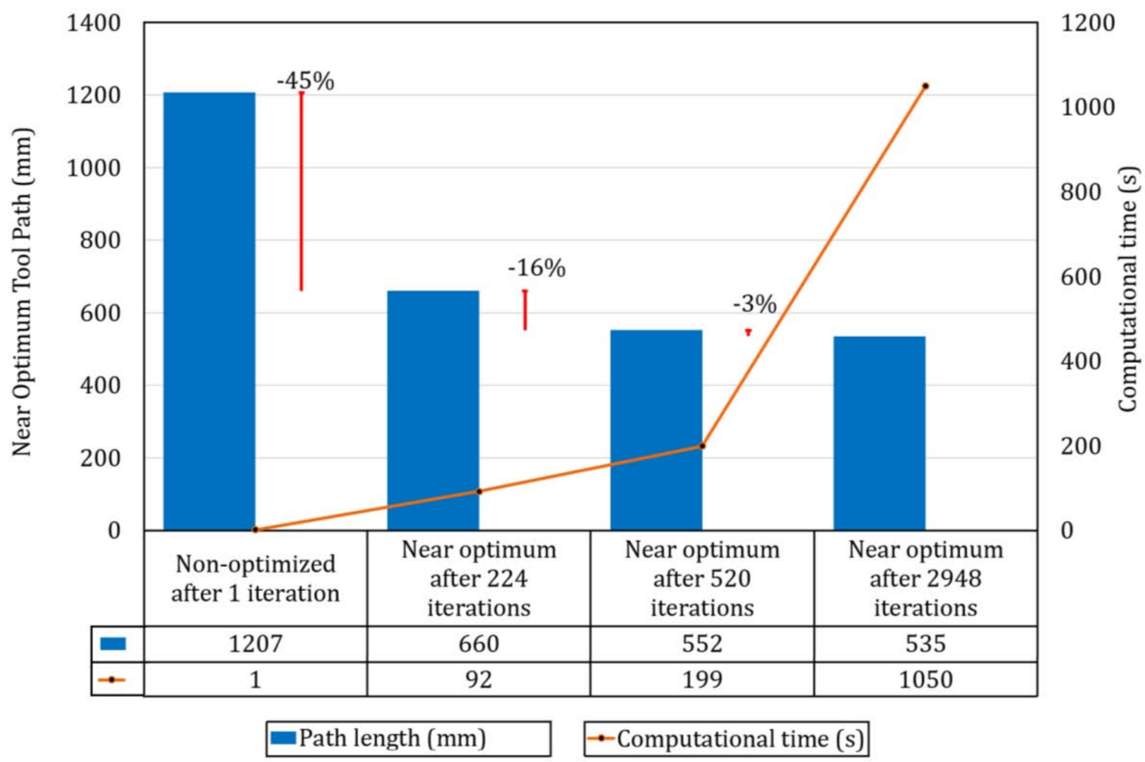


Figure 4.13: Summarise of near optimum tool path and computational time

As it is shown in Figure 4.12 and Figure 4.13, in the first 224 iterations, the proposed algorithm showed 45% reduction in near optimum path length. From iteration number 224 to 520, 16% reduction occurred in the near optimum tool path. The total run time from 1 to 520 is 199 seconds. From iteration 520 to 3000, only 3% reduction occurred in 851 seconds. 3% is not that significant decrease in total tool path to consume 14 minutes for running the algorithm. As a reason, the process can be terminated after 520 iterations, instead of reaching the maximum iterations selected. This 14-minute can be saved and used in manufacturing. As previously discussed, the decision on selecting the maximum number of iterations is left to the judgment of the user based on the time and level of optimality. The best practice is determining a progress limit in the objective function.

In Figure 4.13, the objective function improves only 3% in the last 2500 iteration in around 14 minutes. To fulfill the aforementioned discussion, this time the same example is performed iteratively until the stopping criteria is met. The criterion of the iteration number that is used, is a termination loop in which improvements of near optimum path length is not smaller than 3%. This margin of improvement can change according to the decision of the user. The proposed algorithm terminated in iteration number 559 with the objective function 566 in 158 seconds (see Figure 4.14). In each step the objective function improves more than 3%.

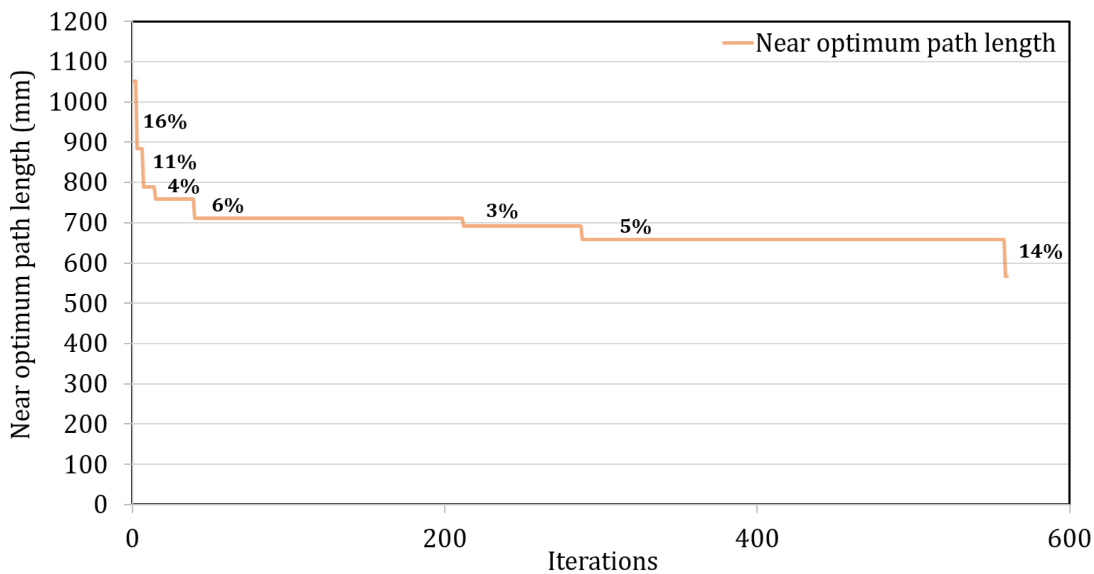


Figure 4.14: Selection of a termination loop in for the case presented in Figure 3.16

4.6 Results and future road map

HSMworks software generates tool path in the least run time, however, the length is considerably higher as shown in Figure 4.15. Imagine a mass production manufacturing system with millions of production units per week. The computational time for HSMWorks occurs only once for the whole production, while the lost time for using a higher tool path length occurs for each of the millions of pieces. The middle column shows the results of the proposed algorithm with a termination loop. As can be seen, the tool path length (566 mm), is almost close to the path length generated by the proposed algorithm with no termination loop, nevertheless, the computational time is much more reasonable. It is not acceptable to perform long computational time, if improvements in the objective function is not significant. Once more, finding the balance between time and level of optimality is important, moving to each way causes sacrifice to the other side.

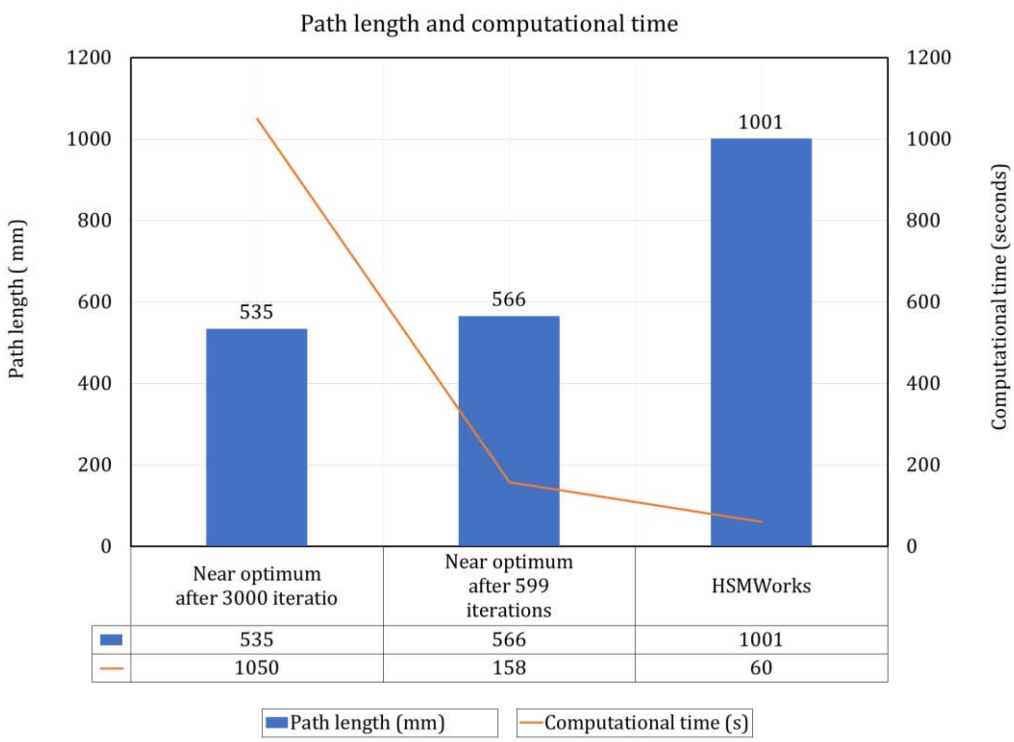


Figure 4.15: Comparison between tool path and computational time for the case presented in Figure 3.16 , tool origin (0,0)

To conclude, the proposed algorithm with a termination loop, shows a perfect balance in computational time and tool path length. Its overall performance considering both time and level of optimality is undeniably better compared to HSMWorks and proposed algorithm with higher iteration number. The results presented in this section prove that the proposed model is able to achieve the shortest tool path when drilling multiple holes on the workpieces with obstacles.

This thesis deals with application of TSP in generating a collision free optimal tool path in drilling operation. Further developing the algorithm to mathematically detect more complex obstacles such as polynomial curves and free form surfaces and generate near optimum tool path can be the focus of future works. Also, it must be noted that the airtime depends not only on the distance travelled, but also on the kinematics of the machine tool especially in 3+2 or 5-axis machining. In such scenarios, the airtime is usually determined by the slowest axis and needs further investigation. The following are the important findings and implications based on the obtained results:

1. The algorithm proposed is capable of optimizing the 14-hole drilling problem. Comparing to the best solution for this particular problem, the tool path generated by the proposed model is only 3.9 % longer (see Table 4.2, columns 3 and 7). Please note that Aziz, et al. [56] did not report the computational time.
2. The new added features of the proposed algorithm including safe tool origin and stopping criteria, avoid high computational time and any human resource intervention. A good selection of safe tool origin not only minimizes the total drilling path length but also eliminates operator’s intervention.
3. The proposed algorithm is capable of providing a shorter collision free path with more than 50% reduction in path length compared to the HSMWorks software. Even the higher the obstacle heights or the larger the number of holes, the improvement in total tool path length reduction is much more significant.
4. The suggestions of the proposed method help manufacturer to reduce time and cost in machining by optimizing the tool path.
5. The algorithm can be developed further as a package to CAD/CAM to minimize the tool airtime length and increase the capability of the machine through suggesting an optimum sequence and a shorter tool retraction height.

4.7 Summary

The problem of optimizing tool paths remains an open field for researchers. Airtime optimization can significantly reduce the machining time and cost, particularly in mass production or production of complex parts. For simple machining processes, generally in industry there is no optimum order, operators can select any sequence according to their skills and knowledge. For complex parts, CAM software helps, however, their generated tool paths are not necessarily optimum. So, application of optimization techniques is advantageous. The problem of optimizing the path length between the holes during drilling can be described as a Travelling Salesman Problem (TSP). In this thesis a new formulation of the TSP method is provided, which, unlike the previously developed methods, includes obstacles as new constraints in generation of collision free tool path in point to point drilling. The new method considers straight and circular obstacles on the tool path.

In the modelling step, nearest neighborhood and local search heuristic algorithms are utilized to perform the optimization in presence of obstacles. The proposed algorithm can suggest a concept in optimization techniques and can be used toward further development of CAM software. It is worthwhile mentioning that the effects of safe tool origin and stopping criteria have also been investigated in this thesis, which is mentioned as future work [15]. The presented case studies, along with the comparison with results from commercial CAM software, confirms the ability of the algorithm in generating an optimum or near optimum collision-free tool path for real-world drilling applications within an acceptable computational time. This research only considers point to point tool paths while the idea of the developed model can also be applied to other processes with continuous tool paths, such as the milling process.

REFERENCES

[1] M. Chen, C. Wang, Q. An, and W. Ming, "Tool path strategy and cutting process monitoring in intelligent machining," *Frontiers of Mechanical Engineering,* vol. 13, no. 2, pp. 232-242, 2018.

[2] R. Dewil, İ. Küçükoğlu, C. Luteyn, and D. Cattrysse, "A critical review of multi-hole drilling path optimization," *Archives of Computational Methods in Engineering,* vol. 26, no. 2, pp. 449-459, 2019.

[3] W. Khan and D. Hayhurst, "Two-and three-dimensional path optimization for production machinery," *J. Manuf. Sci. Eng.,* vol. 122, no. 1, pp. 244-252, 2000.

[4] A. K. Gupta, P. Chandna, and P. Tandon, "Hybrid genetic algorithm for minimizing non productive machining time during 2.5 D milling," *International Journal of Engineering, Science and Technology,* vol. 3, no. 1, 2011.

[5] Y. Altintas, *Manufacturing Automation*. Cambridge University Press, 2012.

[6] C. Oysu and Z. Bingul, "Application of heuristic and hybrid-GASA algorithms to tool-path optimization problem for minimizing airtime during machining," *Engineering Applications of Artificial Intelligence,* vol. 22, no. 3, pp. 389-396, 2009.

[7] R. Ahmad and P. Plapper, "Generation of safe tool-path for 2.5 D milling/drilling machine-tool using 3D ToF sensor," *CIRP Journal of Manufacturing Science and Technology,* vol. 10, pp. 84-91, 2015.

[8] R. Ahmad, S. Tichadou, and J.-Y. Hascoet, "Generation of safe and intelligent tool-paths for multi-axis machine-tools in a dynamic 2D virtual environment," *International Journal of Computer Integrated Manufacturing,* vol. 29, no. 9, pp. 982-995, 2016.

[9] Q. Zhang and S.-R. Li, "Efficient computation of smooth minimum time trajectory for CNC machining," *The International Journal of Advanced Manufacturing Technology,* vol. 68, no. 1-4, pp. 683-692, 2013.

[10] N. Senniappan Karuppusamy and B.-Y. Kang, "Minimizing airtime by optimizing tool path in computer numerical control machine tools with application of A* and genetic algorithms," *Advances in Mechanical Engineering,* vol. 9, no. 12, p. 1687814017737448, 2017.

[11] G. C. Onwubolu, "Optimizing CNC drilling machine operations: traveling salesman problem-differential evolution approach," in *New optimization techniques in engineering*: Springer, 2004, pp. 537-565.

[12] A. Nassehi, W. Essink, and J. Barclay, "Evolutionary algorithms for generation and optimization of tool paths," *CIRP Annals,* vol. 64, no. 1, pp. 455-458, 2015.

[13] N. W. Z. Abidin, M. F. F. A. Rashid, and N. M. Z. N. Mohamed, "Optimization of Multi-holes Drilling Path Using Particle Swarm Optimization," in *Intelligent Manufacturing & Mechatronics*: Springer, 2018, pp. 101-107.

[14] S. Diyaley, A. Aditya, and S. Chakraborty, "Optimization of the multi-hole drilling path sequence for concentric circular patterns," *OPSEARCH,* pp. 1-19, 2020.

[15] C. Huizar, O. Montiel-Ross, R. Sepúlveda, and F. J. D. Delgadillo, "Path planning using clonal selection algorithm," in *Recent Advances on Hybrid Intelligent Systems*: Springer, 2013, pp. 303-312.

[16] I. Lazoglu, C. Manav, and Y. Murtezaoglu, "Tool path optimization for free form surface machining," *CIRP annals,* vol. 58, no. 1, pp. 101-104, 2009.

[17] K. Kiani, M. Sharifi, and M. Shakeri, "Optimization of cutting trajectory to improve manufacturing time in computer numerical control machine using ant colony algorithm," *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture,* vol. 228, no. 7, pp. 811-816, 2014.

[18] M. Hajad, V. Tangwarodomnukun, C. Jaturanonda, and C. Dumkum, "Laser cutting path optimization using simulated annealing with an adaptive large neighborhood search," *The International Journal of Advanced Manufacturing Technology,* vol. 103, no. 1-4, pp. 781-792, 2019.

[19] H. Rico-Garcia, J.-L. Sanchez-Romero, H. M. Gomis, and R. V. Rao, "Parallel implementation of metaheuristics for optimizing tool path computation on CNC machining," *Computers in Industry,* vol. 123, p. 103322, 2020.

[20] D. Pezer, "Efficiency of tool path optimization using genetic algorithm in relation to the optimization achieved with the CAM software," *Procedia Engineering,* vol. 149, pp. 374-379, 2016.

[21] W. Lim, G. Kanagaraj, and S. Ponnambalam, "A hybrid cuckoo search-genetic algorithm for hole-making sequence optimization," *Journal of Intelligent Manufacturing,* vol. 27, no. 2, pp. 417-429, 2014.

[22] M. Ancău, "The optimization of printed circuit board manufacturing by improving the drilling process productivity," *Computers & Industrial Engineering,* vol. 55, no. 2, pp. 279-294, 2008.

[23] N. W. Z. Abidin, M. F. F. Ab Rashid, and N. M. Z. N. Mohamed, "A review of multi-holes drilling path optimization using soft computing approaches," *Archives of Computational Methods in Engineering,* vol. 26, no. 1, pp. 107-118, 2019.

[24] T. Moriwaki, "Multi-functional machine tool," *CIRP annals,* vol. 57, no. 2, pp. 736-749, 2008.

[25] D. Kong, S. Choi, Y. Yasui, S. Pavanaskar, D. Dornfeld, and P. Wright, "Software-based tool path evaluation for environmental sustainability," *Journal of Manufacturing Systems,* vol. 30, no. 4, pp. 241-247, 2011.

[26] G. Onwubolu and M. Clerc, "Optimal path for automated drilling operations by a new heuristic approach using particle swarm optimization," *International Journal of Production Research,* vol. 42, no. 3, pp. 473-491, 2004.

[27] A. T. Abbas, M. F. Aly, and K. Hamza, "Optimum drilling path planning for a rectangular matrix of holes using ant colony optimisation," *International Journal of Production Research,* vol. 49, no. 19, pp. 5877-5891, 2011.

[28] T. L. Schmitz and K. S. Smith, *Machining dynamics*. Springer, 2019.

[29] A. A. Afifi, D. R. Hayhurst, and W. A. Khan, "Non-productive tool path optimisation of multi-tool part programmes," *The International Journal of Advanced Manufacturing Technology,* vol. 55, no. 9-12, pp. 1007-1023, 2011.

[30] C.-S. Lee, J.-H. Lee, D.-S. Kim, E.-Y. Heo, and D.-W. Kim, "A hole-machining process planning system for marine engines," *Journal of manufacturing systems,* vol. 32, no. 1, pp. 114-123, 2013.

[31] W. J. Cook, *In pursuit of the traveling salesman: mathematics at the limits of computation*. Princeton University Press, 2011.

[32] B. F. Voigt, "Der Handlungsreisende, wie er sein soll und was er zu thun hat, um Aufträge zu erhalten und eines glücklichen Erfolgs in seinen Geschäften gewiss zu sein," *Commis-Voageur, Ilmenau*, 1831.

[33] F. J. Díaz-Delgadillo, O. Montiel-Ross, and R. Sepúlveda, "Using immunogenetic algorithms for solving combinatorial optimization problems," in *Recent Advances on Hybrid Intelligent Systems*: Springer, 2013, pp. 273-288.

[34] J. Schneider and S. Kirkpatrick, *Stochastic optimization*. Springer Science & Business Media, 2007.

[35] Y. Marinakis, "Heuristic and metaheuristic algorithms for the traveling salesman problemHeuristic and Metaheuristic Algorithms for the Traveling Salesman Problem," in *Encyclopedia of Optimization*, C. A. Floudas and P. M. Pardalos Eds. Boston, MA: Springer US, 2009, pp. 1498-1506.

[36] A. Kentli and A. F. Alkaya, "Deterministic approach to path optimization problem," *Ozean Journal of Applied Sciences*, vol. 2, no. 2, pp. 149-157, 2009.

[37] R.-M. Aciu and H. Ciocarlie, "G-code optimization algorithm and its application on printed circuit board drilling," in *2014 IEEE 9th IEEE International Symposium on Applied Computational Intelligence and Informatics (SACI)*, 2014: IEEE, pp. 43-47.

[38] K. D. Narooei, R. Ramli, M. N. A. Rahman, F. Iberahim, and J. A. Qudeiri, "Tool routing path optimization for multi-hole drilling based on ant colony optimization," *World Applied Sciences Journal*, vol. 32, no. 9, pp. 1894-1898, 2014.

[39] H. Ghaiebi and M. Solimanpur, "An ant algorithm for optimization of hole-making operations," *Computers & Industrial Engineering*, vol. 52, no. 2, pp. 308-319, 2007.

[40] S. Khalkar, D. Yadav, and A. Singh, "Optimization of hole making operations for sequence precedence constraint," *Int J Innov Emerg Res Eng*, vol. 2, no. 7, pp. 26-31, 2015.

[41] M. Ancău, "The processing time optimization of printed circuit board," *Circuit World*, 2009.

[42] R. J. Linn, J. Liu, and P. S. KOWE, "Efficient heuristics for drilling route optimization in printed circuit board manufacturing," *Journal of Electronics Manufacturing*, vol. 8, no. 02, pp. 127-138, 1998.

[43] M. Saravanan, "Genetic Algorithm for TSP in Optimizing CNC Tool Path."

[44] W. Sheng, N. Xi, M. Song, and Y. Chen, "Robot path planning for dimensional measurement in automotive manufacturing," *J. Manuf. Sci. Eng.*, vol. 127, no. 2, pp. 420-428, 2005.

[45] J. Lee and D.-W. Kim, "An effective initialization method for genetic algorithm-based robot path planning using a directed acyclic graph," *Information Sciences*, vol. 332, pp. 1-18, 2016.

[46] V. Maniezzo, L. M. Gambardella, and F. De Luigi, "Ant colony optimization," in *New Optimization Techniques in Engineering*: Springer, 2004, pp. 101-121.

[47] C. Zhang, F. Han, and W. Zhang, "A cutting sequence optimization method based on tabu search algorithm for complex parts machining," *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture*, vol. 233, no. 3, pp. 745-755, 2019.

[48] M. Bellmore and G. L. Nemhauser, "The traveling salesman problem: a survey," *Operations Research*, vol. 16, no. 3, pp. 538-558, 1968.

[49] J. K. Lenstra and A. R. Kan, "Some simple applications of the travelling salesman problem," *Journal of the Operational Research Society*, vol. 26, no. 4, pp. 717-733, 1975.

[50] E. Lizárraga, O. Castillo, and J. Soria, "A method to solve the traveling salesman problem using ant colony optimization variants with ant set partitioning," in *Recent Advances on Hybrid Intelligent Systems*: Springer, 2013, pp. 237-246.

[51] D. S. Johnson and L. A. McGeoch, "The traveling salesman problem: A case study in local optimization," *Local search in combinatorial optimization*, vol. 1, no. 1, pp. 215-310, 1997.

[52] G. Gutin and A. P. Punnen, *The traveling salesman problem and its variations*. Springer Science & Business Media, 2006.

[53] G. Reinelt, *The traveling salesman: computational solutions for TSP applications*. Springer, 2003.

[54] G. Y. Zhu, "Drilling path optimization based on swarm intelligent algorithm," in *2006 IEEE International Conference on Robotics and Biomimetics*, 2006: IEEE, pp. 193-196.

[55] G.-Y. Zhu and W.-B. Zhang, "Drilling path optimization by the particle swarm optimization algorithm with global convergence characteristics," *International Journal of Production Research*, vol. 46, no. 8, pp. 2299-2311, 2008.

[56] N. H. A. Aziz, Z. Ibrahim, N. A. Ab Aziz, Z. M. Yusof, and M. S. Mohamad, "Single-solution simulated Kalman filter algorithm for routing in printed circuit board drilling process," in *Intelligent Manufacturing & Mechatronics*: Springer, 2018, pp. 649-655.

[57] J. Gu and X. Huang, "Efficient local search with search space smoothing: A case study of the traveling salesman problem (TSP)," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 24, no. 5, pp. 728-735, 1994.

[58] G. C. Onwubolu and B. Babu, *New Optimization Techniques in Engineering* (Studies in Fuzziness and Soft Computing). Springer, 2004.

[59] L. Du and R. He, "Combining nearest neighbor search with tabu search for large-scale vehicle routing problem," *Physics Procedia*, vol. 25, pp. 1536-1546, 2012.

[60] G. Reinelt, *The traveling salesman: computational solutions for TSP applications*. Springer-Verlag, 1994.

[61] A. P. Punnen, G. Gutin, and A. P. Punnen, "The traveling salesman problem and its variations," 2007.

Appendix A: Mathematical proof of minimum distance in circle

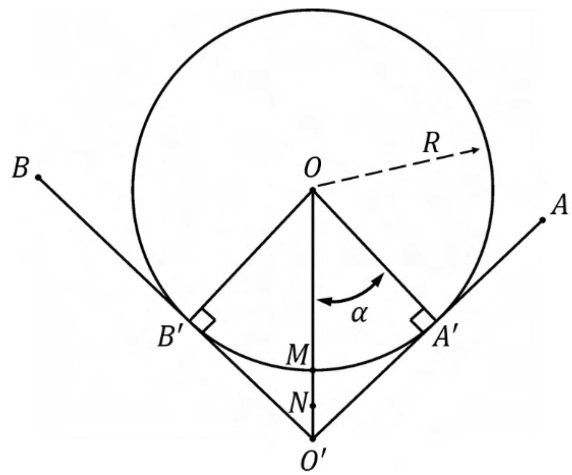


Figure A.1: Schematic circle that used for mathematical proof of tangent line

Angles are in radian:

I: $A'M = \alpha$R

SAS (side-angle-side) Two sides and the angle between them are congruent:

$$\Delta OA'O' \equiv \Delta OB'O' \quad \rightarrow A'B' = 2A'M$$

II: $\tan\alpha = \frac{O'A'}{OA'} \quad \rightarrow \quad O'A' = OA'.\tan\alpha$

III: From I , II : $\frac{A'M}{O'A'} = \frac{\alpha R}{OA'.\tan\alpha} \xrightarrow{OA'=R} \frac{A'M}{O'A'} = \frac{\alpha R}{Rtan\alpha} \xrightarrow{\frac{\alpha}{tan\alpha}<1} \frac{A'M}{O'A'} < 1 \rightarrow \; A'M < O'A'$

IV: Proof in a similar way: $B'M < O'B'$

From III, IV: $A'M + B'M < O'A' + O'B' \rightarrow \; A'B' < O'A' + O'B'$

Any other point chosen on the $O'M$ line like N, the same proof as above shows that:

$$A'B' < NA' + NB'$$

Appendix B: Copyright Permission Letter for Figure 2.6

3/25/2021 RightsLink Printable License

ELSEVIER LICENSE
TERMS AND CONDITIONS

Mar 25, 2021

This Agreement between University of Ontario Institute of Technology -- Zahra Khodabakhshi ("You") and Elsevier ("Elsevier") consists of your license details and the terms and conditions provided by Elsevier and Copyright Clearance Center.

License Number	5036020939362
License date	Mar 25, 2021
Licensed Content Publisher	Elsevier
Licensed Content Publication	Journal of Manufacturing Systems
Licensed Content Title	A hole-machining process planning system for marine engines
Licensed Content Author	Cheol-Soo Lee,Jae-Hyun Lee,Dong-Soo Kim,Eun-Young Heo,Dong-Won Kim
Licensed Content Date	Jan 1, 2013
Licensed Content Volume	32
Licensed Content Issue	1
Licensed Content Pages	10
Start Page	114
End Page	123

https://s100.copyright.com/AppDispatchServlet 1/8

Appendix

Type of Use	reuse in a thesis/dissertation
Portion	figures/tables/illustrations
Number of figures/tables/illustrations	1
Format	both print and electronic
Are you the author of this Elsevier article?	No
Will you be translating?	No
Title	tentative title " application of travelling salesman problem in tool path optimization"
Institution name	University of Ontario Institute of Technology
Expected presentation date	Apr 2021
Portions	Fig.6 page 4
Requestor Location	Zahra Khodabakhshi 2000 Simcoe Street North Oshawa, ON L1G 0C5 Canada Attn: University of Ontario Institute of Technology (Ontario Tech University)
Publisher Tax ID	GB 494 6272 12
Total	0.00 CAD
Terms and Conditions	

INTRODUCTION

1. The publisher for this copyrighted material is Elsevier. By clicking "accept" in connection with completing this licensing transaction, you agree that the following terms and conditions apply to this transaction (along with the Billing and Payment terms and conditions established by Copyright Clearance Center, Inc. ("CCC"), at the time that you opened your Rightslink account and that are available at any time at http://myaccount.copyright.com).

GENERAL TERMS

2. Elsevier hereby grants you permission to reproduce the aforementioned material subject to the terms and conditions indicated.

3. Acknowledgement: If any part of the material to be used (for example, figures) has appeared in our publication with credit or acknowledgement to another source, permission must also be sought from that source. If such permission is not obtained then that material may not be included in your publication/copies. Suitable acknowledgement to the source must be made, either as a footnote or in a reference list at the end of your publication, as follows:

"Reprinted from Publication title, Vol /edition number, Author(s), Title of article / title of chapter, Pages No., Copyright (Year), with permission from Elsevier [OR APPLICABLE SOCIETY COPYRIGHT OWNER]." Also Lancet special credit - "Reprinted from The Lancet, Vol. number, Author(s), Title of article, Pages No., Copyright (Year), with permission from Elsevier."

4. Reproduction of this material is confined to the purpose and/or media for which permission is hereby given.

5. Altering/Modifying Material: Not Permitted. However figures and illustrations may be altered/adapted minimally to serve your work. Any other abbreviations, additions, deletions and/or any other alterations shall be made only with prior written authorization of Elsevier Ltd. (Please contact Elsevier's permissions helpdesk here). No modifications can be made to any Lancet figures/tables and they must be reproduced in full.

6. If the permission fee for the requested use of our material is waived in this instance, please be advised that your future requests for Elsevier materials may attract a fee.

7. Reservation of Rights: Publisher reserves all rights not specifically granted in the combination of (i) the license details provided by you and accepted in the course of this licensing transaction, (ii) these terms and conditions and (iii) CCC's Billing and Payment terms and conditions.

8. License Contingent Upon Payment: While you may exercise the rights licensed immediately upon issuance of the license at the end of the licensing process for the transaction, provided that you have disclosed complete and accurate details of your proposed use, no license is finally effective unless and until full payment is received from you (either by publisher or by CCC) as provided in CCC's Billing and Payment terms and conditions. If full payment is not received on a timely basis, then any license preliminarily granted shall be deemed automatically revoked and shall be void as if never granted. Further, in the event that you breach any of these terms and conditions or any of CCC's Billing and Payment terms and conditions, the license is automatically revoked and shall be void as if never granted. Use of materials as described in a revoked license, as well as any use of the materials beyond the scope of an unrevoked license, may constitute copyright infringement and publisher reserves the right to take any and all action to protect its copyright in the materials.

9. Warranties: Publisher makes no representations or warranties with respect to the licensed material.

10. Indemnity: You hereby indemnify and agree to hold harmless publisher and CCC, and their respective officers, directors, employees and agents, from and against any and all claims arising out of your use of the licensed material other than as specifically authorized pursuant to this license.

11. No Transfer of License: This license is personal to you and may not be sublicensed, assigned, or transferred by you to any other person without publisher's written permission.

12. No Amendment Except in Writing: This license may not be amended except in a writing signed by both parties (or, in the case of publisher, by CCC on publisher's behalf).

13. Objection to Contrary Terms: Publisher hereby objects to any terms contained in any purchase order, acknowledgment, check endorsement or other writing prepared by you, which terms are inconsistent with these terms and conditions or CCC's Billing and Payment terms and conditions. These terms and conditions, together with CCC's Billing and Payment terms and conditions (which are incorporated herein), comprise the entire agreement between you and publisher (and CCC) concerning this licensing transaction. In the event of any conflict between your obligations established by these terms and conditions and those established by CCC's Billing and Payment terms and conditions, these terms and conditions shall control.

14. Revocation: Elsevier or Copyright Clearance Center may deny the permissions described in this License at their sole discretion, for any reason or no reason, with a full refund payable to you. Notice of such denial will be made using the contact information provided by you. Failure to receive such notice will not alter or invalidate the denial. In no event will Elsevier or Copyright Clearance Center be responsible or liable for any costs, expenses or damage incurred by you as a result of a denial of your permission request, other than a refund of the amount(s) paid by you to Elsevier and/or Copyright Clearance Center for denied permissions.

LIMITED LICENSE

The following terms and conditions apply only to specific license types:

15. **Translation**: This permission is granted for non-exclusive world **English** rights only unless your license was granted for translation rights. If you licensed translation rights you may only translate this content into the languages you requested. A professional translator must perform all translations and reproduce the content word for word preserving the integrity of the article.

16. **Posting licensed content on any Website**: The following terms and conditions apply as follows: Licensing material from an Elsevier journal: All content posted to the web site must maintain the copyright information line on the bottom of each image; A hyper-text must be included to the Homepage of the journal from which you are licensing at http://www.sciencedirect.com/science/journal/xxxxx or the Elsevier homepage for books at http://www.elsevier.com; Central Storage: This license does not include permission for a scanned version of the material to be stored in a central repository such as that provided by Heron/XanEdu.

Licensing material from an Elsevier book: A hyper-text link must be included to the Elsevier homepage at http://www.elsevier.com . All content posted to the web site must maintain the copyright information line on the bottom of each image.

Posting licensed content on Electronic reserve: In addition to the above the following clauses are applicable: The web site must be password-protected and made available only to bona fide students registered on a relevant course. This permission is granted for 1 year only. You may obtain a new license for future website posting.

17. **For journal authors:** the following clauses are applicable in addition to the above:

Preprints:

A preprint is an author's own write-up of research results and analysis, it has not been peer-reviewed, nor has it had any other value added to it by a publisher (such as formatting, copyright, technical enhancement etc.).

Authors can share their preprints anywhere at any time. Preprints should not be added to or enhanced in any way in order to appear more like, or to substitute for, the final versions of articles however authors can update their preprints on arXiv or RePEc with their Accepted Author Manuscript (see below).

If accepted for publication, we encourage authors to link from the preprint to their formal publication via its DOI. Millions of researchers have access to the formal publications on ScienceDirect, and so links will help users to find, access, cite and use the best available version. Please note that Cell Press, The Lancet and some society-owned have different preprint policies. Information on these policies is available on the journal homepage.

Accepted Author Manuscripts: An accepted author manuscript is the manuscript of an article that has been accepted for publication and which typically includes author-incorporated changes suggested during submission, peer review and editor-author communications.

Authors can share their accepted author manuscript:

- immediately
  - via their non-commercial person homepage or blog
  - by updating a preprint in arXiv or RePEc with the accepted manuscript
  - via their research institute or institutional repository for internal institutional uses or as part of an invitation-only research collaboration work-group
  - directly by providing copies to their students or to research collaborators for their personal use
  - for private scholarly sharing as part of an invitation-only work group on commercial sites with which Elsevier has an agreement
- After the embargo period
  - via non-commercial hosting platforms such as their institutional repository
  - via commercial sites with which Elsevier has an agreement

In all cases accepted manuscripts should:

- link to the formal publication via its DOI
- bear a CC-BY-NC-ND license - this is easy to do
- if aggregated with other manuscripts, for example in a repository or other site, be shared in alignment with our hosting policy not be added to or enhanced in any way to appear more like, or to substitute for, the published journal article.

Published journal article (JPA): A published journal article (PJA) is the definitive final record of published research that appears or will appear in the journal and embodies all value-adding publishing activities including peer review co-ordination, copy-editing, formatting, (if relevant) pagination and online enrichment.

Policies for sharing publishing journal articles differ for subscription and gold open access articles:

Subscription Articles: If you are an author, please share a link to your article rather than the full-text. Millions of researchers have access to the formal publications on ScienceDirect, and so links will help your users to find, access, cite, and use the best available version.

Theses and dissertations which contain embedded PJAs as part of the formal submission can be posted publicly by the awarding institution with DOI links back to the formal publications on ScienceDirect.

If you are affiliated with a library that subscribes to ScienceDirect you have additional private sharing rights for others' research accessed under that agreement. This includes use for classroom teaching and internal training at the institution (including use in course packs and courseware programs), and inclusion of the article for grant funding purposes.

Gold Open Access Articles: May be shared according to the author-selected end-user license and should contain a CrossMark logo, the end user license, and a DOI link to the formal publication on ScienceDirect.

Please refer to Elsevier's posting policy for further information.

18. **For book authors** the following clauses are applicable in addition to the above: Authors are permitted to place a brief summary of their work online only. You are not allowed to download and post the published electronic version of your chapter, nor may you scan the printed edition to create an electronic version. **Posting to a repository:** Authors are permitted to post a summary of their chapter only in their institution's repository.

19. **Thesis/Dissertation**: If your license is for use in a thesis/dissertation your thesis may be submitted to your institution in either print or electronic form. Should your thesis be published commercially, please reapply for permission. These requirements include permission for the Library and Archives of Canada to supply single copies, on demand, of the complete thesis and include permission for Proquest/UMI to supply single copies, on demand, of the complete thesis. Should your thesis be published commercially, please reapply for permission. Theses and dissertations which contain embedded PJAs as part of the formal submission can be posted publicly by the awarding institution with DOI links back to the formal publications on ScienceDirect.

Elsevier Open Access Terms and Conditions

You can publish open access with Elsevier in hundreds of open access journals or in nearly 2000 established subscription journals that support open access publishing. Permitted third party re-use of these open access articles is defined by the author's choice of Creative Commons user license. See our open access license policy for more information.

Terms & Conditions applicable to all Open Access articles published with Elsevier:

Any reuse of the article must not represent the author as endorsing the adaptation of the article nor should the article be modified in such a way as to damage the author's honour or reputation. If any changes have been made, such changes must be clearly indicated.

The author(s) must be appropriately credited and we ask that you include the end user license and a DOI link to the formal publication on ScienceDirect.

If any part of the material to be used (for example, figures) has appeared in our publication with credit or acknowledgement to another source it is the responsibility of the user to ensure their reuse complies with the terms and conditions determined by the rights holder.

Additional Terms & Conditions applicable to each Creative Commons user license:

CC BY: The CC-BY license allows users to copy, to create extracts, abstracts and new works from the Article, to alter and revise the Article and to make commercial use of the Article (including reuse and/or resale of the Article by commercial entities), provided the user gives appropriate credit (with a link to the formal publication through the relevant DOI), provides a link to the license, indicates if changes were made and the licensor is not represented as endorsing the use made of the work. The full details of the license are available at http://creativecommons.org/licenses/by/4.0.

CC BY NC SA: The CC BY-NC-SA license allows users to copy, to create extracts, abstracts and new works from the Article, to alter and revise the Article, provided this is not done for commercial purposes, and that the user gives appropriate credit (with a link to the formal publication through the relevant DOI), provides a link to the license, indicates if changes were made and the licensor is not represented as endorsing the use made of the work. Further, any new works must be made available on the same conditions. The full details of the license are available at http://creativecommons.org/licenses/by-nc-sa/4.0.

CC BY NC ND: The CC BY-NC-ND license allows users to copy and distribute the Article, provided this is not done for commercial purposes and further does not permit distribution of the Article if it is changed or edited in any way, and provided the user gives appropriate credit (with a link to the formal publication through the relevant DOI), provides a link to the license, and that the licensor is not represented as endorsing the use made of the work. The full details of the license are available at http://creativecommons.org/licenses/by-nc-nd/4.0. Any commercial reuse of Open Access articles published with a CC BY NC SA or CC BY NC ND license requires permission from Elsevier and will be subject to a fee.

Commercial reuse includes:

- Associating advertising with the full text of the Article
- Charging fees for document delivery or access
- Article aggregation
- Systematic distribution via e-mail lists or share buttons

Posting or linking by commercial companies for use by customers of those companies.

20. **Other Conditions**:

v1.10

Questions? customercare@copyright.com or +1-855-239-3415 (toll free in the US) or +1-978-646-2777.

Appendix C: Copyright Permission Letter for Figure 2.8

3/25/2021 RightsLink Printable License

ELSEVIER LICENSE
TERMS AND CONDITIONS

Mar 25, 2021

This Agreement between University of Ontario Institute of Technology -- Zahra Khodabakhshi ("You") and Elsevier ("Elsevier") consists of your license details and the terms and conditions provided by Elsevier and Copyright Clearance Center.

License Number	5036020703445
License date	Mar 25, 2021
Licensed Content Publisher	Elsevier
Licensed Content Publication	Information Sciences
Licensed Content Title	An effective initialization method for genetic algorithm-based robot path planning using a directed acyclic graph
Licensed Content Author	Jaesung Lee,Dae-Won Kim
Licensed Content Date	Mar 1, 2016
Licensed Content Volume	332
Licensed Content Issue	n/a
Licensed Content Pages	18
Start Page	1
End Page	18

https://s100.copyright.com/AppDispatchServlet 1/8

Appendix

Type of Use	reuse in a thesis/dissertation
Portion	figures/tables/illustrations
Number of figures/tables/illustrations	2
Format	both print and electronic
Are you the author of this Elsevier article?	No
Will you be translating?	No
Title	tentative title " application of travelling salesman problem in tool path optimization"
Institution name	University of Ontario Institute of Technology
Expected presentation date	Apr 2021
Portions	Fig.1 (a) and (b) on page 3
Requestor Location	Zahra Khodabakhshi 2000 Simcoe Street North Oshawa, ON L1G 0C5 Canada Attn: University of Ontario Institute of Technology (Ontario Tech University)
Publisher Tax ID	GB 494 6272 12
Total	0.00 CAD
Terms and Conditions	

INTRODUCTION

1. The publisher for this copyrighted material is Elsevier. By clicking "accept" in connection with completing this licensing transaction, you agree that the following terms and conditions apply to this transaction (along with the Billing and Payment terms and conditions established by Copyright Clearance Center, Inc. ("CCC"), at the time that you opened your Rightslink account and that are available at any time at http://myaccount.copyright.com).

GENERAL TERMS

2. Elsevier hereby grants you permission to reproduce the aforementioned material subject to the terms and conditions indicated.

3. Acknowledgement: If any part of the material to be used (for example, figures) has appeared in our publication with credit or acknowledgement to another source, permission must also be sought from that source. If such permission is not obtained then that material may not be included in your publication/copies. Suitable acknowledgement to the source must be made, either as a footnote or in a reference list at the end of your publication, as follows:

"Reprinted from Publication title, Vol /edition number, Author(s), Title of article / title of chapter, Pages No., Copyright (Year), with permission from Elsevier [OR APPLICABLE SOCIETY COPYRIGHT OWNER]." Also Lancet special credit - "Reprinted from The Lancet, Vol. number, Author(s), Title of article, Pages No., Copyright (Year), with permission from Elsevier."

4. Reproduction of this material is confined to the purpose and/or media for which permission is hereby given.

5. Altering/Modifying Material: Not Permitted. However figures and illustrations may be altered/adapted minimally to serve your work. Any other abbreviations, additions, deletions and/or any other alterations shall be made only with prior written authorization of Elsevier Ltd. (Please contact Elsevier's permissions helpdesk here). No modifications can be made to any Lancet figures/tables and they must be reproduced in full.

6. If the permission fee for the requested use of our material is waived in this instance, please be advised that your future requests for Elsevier materials may attract a fee.

7. Reservation of Rights: Publisher reserves all rights not specifically granted in the combination of (i) the license details provided by you and accepted in the course of this licensing transaction, (ii) these terms and conditions and (iii) CCC's Billing and Payment terms and conditions.

8. License Contingent Upon Payment: While you may exercise the rights licensed immediately upon issuance of the license at the end of the licensing process for the transaction, provided that you have disclosed complete and accurate details of your proposed use, no license is finally effective unless and until full payment is received from you (either by publisher or by CCC) as provided in CCC's Billing and Payment terms and conditions. If full payment is not received on a timely basis, then any license preliminarily granted shall be deemed automatically revoked and shall be void as if never granted. Further, in the event that you breach any of these terms and conditions or any of CCC's Billing and Payment terms and conditions, the license is automatically revoked and shall be void as if never granted. Use of materials as described in a revoked license, as well as any use of the materials beyond the scope of an unrevoked license, may constitute copyright infringement and publisher reserves the right to take any and all action to protect its copyright in the materials.

9. Warranties: Publisher makes no representations or warranties with respect to the licensed material.

10. Indemnity: You hereby indemnify and agree to hold harmless publisher and CCC, and their respective officers, directors, employees and agents, from and against any and all claims arising out of your use of the licensed material other than as specifically authorized pursuant to this license.

11. No Transfer of License: This license is personal to you and may not be sublicensed, assigned, or transferred by you to any other person without publisher's written permission.

12. No Amendment Except in Writing: This license may not be amended except in a writing signed by both parties (or, in the case of publisher, by CCC on publisher's behalf).

13. Objection to Contrary Terms: Publisher hereby objects to any terms contained in any purchase order, acknowledgment, check endorsement or other writing prepared by you, which terms are inconsistent with these terms and conditions or CCC's Billing and Payment terms and conditions. These terms and conditions, together with CCC's Billing and Payment terms and conditions (which are incorporated herein), comprise the entire agreement between you and publisher (and CCC) concerning this licensing transaction. In the event of any conflict between your obligations established by these terms and conditions and those established by CCC's Billing and Payment terms and conditions, these terms and conditions shall control.

14. Revocation: Elsevier or Copyright Clearance Center may deny the permissions described in this License at their sole discretion, for any reason or no reason, with a full refund payable to you. Notice of such denial will be made using the contact information provided by you. Failure to receive such notice will not alter or invalidate the denial. In no event will Elsevier or Copyright Clearance Center be responsible or liable for any costs, expenses or damage incurred by you as a result of a denial of your permission request, other than a refund of the amount(s) paid by you to Elsevier and/or Copyright Clearance Center for denied permissions.

LIMITED LICENSE

The following terms and conditions apply only to specific license types:

15. **Translation**: This permission is granted for non-exclusive world **English** rights only unless your license was granted for translation rights. If you licensed translation rights you may only translate this content into the languages you requested. A professional translator must perform all translations and reproduce the content word for word preserving the integrity of the article.

16. **Posting licensed content on any Website**: The following terms and conditions apply as follows: Licensing material from an Elsevier journal: All content posted to the web site must maintain the copyright information line on the bottom of each image; A hyper-text must be included to the Homepage of the journal from which you are licensing at http://www.sciencedirect.com/science/journal/xxxxx or the Elsevier homepage for books at http://www.elsevier.com; Central Storage: This license does not include permission for a scanned version of the material to be stored in a central repository such as that provided by Heron/XanEdu.

Licensing material from an Elsevier book: A hyper-text link must be included to the Elsevier homepage at http://www.elsevier.com . All content posted to the web site must maintain the copyright information line on the bottom of each image.

Posting licensed content on Electronic reserve: In addition to the above the following clauses are applicable: The web site must be password-protected and made available only to bona fide students registered on a relevant course. This permission is granted for 1 year only. You may obtain a new license for future website posting.

17. **For journal authors:** the following clauses are applicable in addition to the above:

Preprints:

A preprint is an author's own write-up of research results and analysis, it has not been peer-reviewed, nor has it had any other value added to it by a publisher (such as formatting, copyright, technical enhancement etc.).

Authors can share their preprints anywhere at any time. Preprints should not be added to or enhanced in any way in order to appear more like, or to substitute for, the final versions of articles however authors can update their preprints on arXiv or RePEc with their Accepted Author Manuscript (see below).

If accepted for publication, we encourage authors to link from the preprint to their formal publication via its DOI. Millions of researchers have access to the formal publications on ScienceDirect, and so links will help users to find, access, cite and use the best available version. Please note that Cell Press, The Lancet and some society-owned have different preprint policies. Information on these policies is available on the journal homepage.

Accepted Author Manuscripts: An accepted author manuscript is the manuscript of an article that has been accepted for publication and which typically includes author-incorporated changes suggested during submission, peer review and editor-author communications.

Authors can share their accepted author manuscript:

- immediately
    - via their non-commercial person homepage or blog
    - by updating a preprint in arXiv or RePEc with the accepted manuscript
    - via their research institute or institutional repository for internal institutional uses or as part of an invitation-only research collaboration work-group
    - directly by providing copies to their students or to research collaborators for their personal use
    - for private scholarly sharing as part of an invitation-only work group on commercial sites with which Elsevier has an agreement
- After the embargo period
    - via non-commercial hosting platforms such as their institutional repository
    - via commercial sites with which Elsevier has an agreement

In all cases accepted manuscripts should:

- link to the formal publication via its DOI
- bear a CC-BY-NC-ND license - this is easy to do
- if aggregated with other manuscripts, for example in a repository or other site, be shared in alignment with our hosting policy not be added to or enhanced in any way to appear more like, or to substitute for, the published journal article.

Published journal article (JPA): A published journal article (PJA) is the definitive final record of published research that appears or will appear in the journal and embodies all value-adding publishing activities including peer review co-ordination, copy-editing, formatting, (if relevant) pagination and online enrichment.

Policies for sharing publishing journal articles differ for subscription and gold open access articles:

Subscription Articles: If you are an author, please share a link to your article rather than the full-text. Millions of researchers have access to the formal publications on ScienceDirect, and so links will help your users to find, access, cite, and use the best available version.

Theses and dissertations which contain embedded PJAs as part of the formal submission can be posted publicly by the awarding institution with DOI links back to the formal publications on ScienceDirect.

If you are affiliated with a library that subscribes to ScienceDirect you have additional private sharing rights for others' research accessed under that agreement. This includes use for classroom teaching and internal training at the institution (including use in course packs and courseware programs), and inclusion of the article for grant funding purposes.

Gold Open Access Articles: May be shared according to the author-selected end-user license and should contain a CrossMark logo, the end user license, and a DOI link to the formal publication on ScienceDirect.

Please refer to Elsevier's posting policy for further information.

18. **For book authors** the following clauses are applicable in addition to the above: Authors are permitted to place a brief summary of their work online only. You are not allowed to download and post the published electronic version of your chapter, nor may you scan the printed edition to create an electronic version. **Posting to a repository:** Authors are permitted to post a summary of their chapter only in their institution's repository.

19. **Thesis/Dissertation**: If your license is for use in a thesis/dissertation your thesis may be submitted to your institution in either print or electronic form. Should your thesis be published commercially, please reapply for permission. These requirements include permission for the Library and Archives of Canada to supply single copies, on demand, of the complete thesis and include permission for Proquest/UMI to supply single copies, on demand, of the complete thesis. Should your thesis be published commercially, please reapply for permission. Theses and dissertations which contain embedded PJAs as part of the formal submission can be posted publicly by the awarding institution with DOI links back to the formal publications on ScienceDirect.

Elsevier Open Access Terms and Conditions

You can publish open access with Elsevier in hundreds of open access journals or in nearly 2000 established subscription journals that support open access publishing. Permitted third party re-use of these open access articles is defined by the author's choice of Creative Commons user license. See our open access license policy for more information.

Terms & Conditions applicable to all Open Access articles published with Elsevier:

Any reuse of the article must not represent the author as endorsing the adaptation of the article nor should the article be modified in such a way as to damage the author's honour or reputation. If any changes have been made, such changes must be clearly indicated.

The author(s) must be appropriately credited and we ask that you include the end user license and a DOI link to the formal publication on ScienceDirect.

If any part of the material to be used (for example, figures) has appeared in our publication with credit or acknowledgement to another source it is the responsibility of the user to ensure their reuse complies with the terms and conditions determined by the rights holder.

Additional Terms & Conditions applicable to each Creative Commons user license:

CC BY: The CC-BY license allows users to copy, to create extracts, abstracts and new works from the Article, to alter and revise the Article and to make commercial use of the Article (including reuse and/or resale of the Article by commercial entities), provided the user gives appropriate credit (with a link to the formal publication through the relevant DOI), provides a link to the license, indicates if changes were made and the licensor is not represented as endorsing the use made of the work. The full details of the license are available at http://creativecommons.org/licenses/by/4.0.

CC BY NC SA: The CC BY-NC-SA license allows users to copy, to create extracts, abstracts and new works from the Article, to alter and revise the Article, provided this is not done for commercial purposes, and that the user gives appropriate credit (with a link to the formal publication through the relevant DOI), provides a link to the license, indicates if changes were made and the licensor is not represented as endorsing the use made of the work. Further, any new works must be made available on the same conditions. The full details of the license are available at http://creativecommons.org/licenses/by-nc-sa/4.0.

CC BY NC ND: The CC BY-NC-ND license allows users to copy and distribute the Article, provided this is not done for commercial purposes and further does not permit distribution of the Article if it is changed or edited in any way, and provided the user gives appropriate credit (with a link to the formal publication through the relevant DOI), provides a link to the license, and that the licensor is not represented as endorsing the use made of the work. The full details of the license are available at http://creativecommons.org/licenses/by-nc-nd/4.0. Any commercial reuse of Open Access articles published with a CC BY NC SA or CC BY NC ND license requires permission from Elsevier and will be subject to a fee.

Commercial reuse includes:

- Associating advertising with the full text of the Article
- Charging fees for document delivery or access
- Article aggregation
- Systematic distribution via e-mail lists or share buttons

Posting or linking by commercial companies for use by customers of those companies.

20. **Other Conditions**:

v1.10

Questions? customercare@copyright.com or +1-855-239-3415 (toll free in the US) or +1-978-646-2777.