August 2020

SOLVING PROCESS PLANNING AND SCHEDULING PROBLEMS USING THE CONCEPT OF MAXIMUM WEIGHTED INDEPENDENT SET

Kai Sun
Syracuse University

Abstract

Process planning and scheduling (PPS) is an essential and practical topic but a very intractable problem in manufacturing systems. Many research studies use iterative methods to solve such problems; however, they cannot achieve satisfactory results in both quality and computational speed. Other studies formulate scheduling problems as a graph coloring problem (GCP) or its extensions, but these formulations are limited to certain types of scheduling problems. In this dissertation, we propose a novel approach to formulate a general type of the PPS problem with resource allocation and process planning integrated towards a typical objective, minimizing the makespan. The PPS problem is formulated into an undirected weighted conflicting graph, where nodes represent operations and their resources; edges represent constraints, and weight factors are guidelines for the node selection at each time slot. Then, the Maximum Weighted Independent Set (MWIS) problem, which considers a graph with weights assigned to nodes and seeks to discover the "heaviest" independent set, that is, a set of nodes with maximum total weight so that no two nodes in the set are connected by an edge, can be solved to find the best set of operations with their desired resources for each discrete time slot.

This proposed approach solves the PPS problem directly (a direct method in computational mathematics context). We establish that the proposed approach always returns a feasible optimum or near-optimum solution to the PPS problem.

The performance of the proposed approach for the PPS problem depends on the accuracy and computational speed of solving the MWIS problem. We propose a divide-and-conquer algorithm structure with relatively low complexity for solving the MWIS problem. An exact MWIS algorithm and an All Maximal Independent Set Listing (AMISL) algorithm are developed based on this algorithm structure. The proposed algorithm structure can also be used to compose the exact MWIS algorithm with existing approximation MWIS algorithms. This is an effective way to improve the accuracy of existing approximation MWIS algorithms or improve the computational speed of the exact MWIS algorithm.

All eight algorithms for the MWIS problem, the exact MWIS algorithm, the AMISL algorithm, two approximation algorithms from the literature, and four composed algorithms, are tested on the test instances based on the PPS application environment. The different configurations of the proposed approach for solving the PPS problem are tested on a real-world PPS example and further designated test instances to evaluate the scalability, accuracy, and robustness.

SOLVING PROCESS PLANNING AND SCHEDULING PROBLEMS USING THE CONCEPT OF MAXIMUM WEIGHTED INDEPENDENT SET

by

Kai Sun

B.S., Hefei University of Technology, 2013
M.S., Syracuse University, 2015

Dissertation
Submitted in partial fulfillment of the requirements for the degree of
Doctor of Philosophy in Mechanical and Aerospace Engineering.

Syracuse University
August 2020

Acknowledgments

I would like to express my deepest gratitude to my advisor, Prof. Utpal Roy. His wisdom and vision guided me and prepared me to be a better researcher. His patience and encouragement carried me on through difficult times. This dissertation could not have been written without Prof. Roy, who not only helped me sort out the technical details, but also carefully corrected the language and phrasing. I appreciate Prof. Roy to provide all the opportunities for academia and industrial exposure.

I would like to thank Prof. John F. Dannenhoffer, III, for his guidance since last summer. His attitude towards science and vision impressed me and guided me in the right research direction. His passion and enthusiasm towards research motivated and encouraged me to accomplish my Ph.D. study. His valuable comments have been extremely helpful for improving my research.

I would like to thank Prof. Jack Graver for bringing me into the world of Graph Theory. His knowledgeable lectures built a strong background and inspired me to develop new ideas. His guidance and suggestions helped me build the bridge between mathematics and engineering.

I would like to thank Prof. Xiyuan Liu for serving in my Ph.D. committee from the beginning to the end. Her knowledge and experiments are great fortune for the challenges in the past years and the future. I would like to thank Prof. Young B. Moon for serving in my Ph.D. committee. I learned a lot by attending his knowledgeable lectures, and his rigorous attitude to science sets up the model for myself.

During the years of my Ph.D. study at Syracuse University, I received generous help from fellow students and close friends. Especially my best friends Dr. Bicheng Zhu and Dr. Yueming Song, who were always there to help me and keep my morale up. I would like to thank all past and current members in Prof. Roy's lab: Dr. Heng Zhang, Dr. Yunpeng Li, Dr. Hang Yin, Mr. Omar Yaman, and Mr. Cheng Li for many helpful discussions. Special thanks go to Ms. Rui Hou, Dr. Tianji Yang, and Mr. Zhuhui He for their friendship and support over the distance. I also want to thank all my friends for their help and companionship in the past years.

Finally, I would like to thank my parents: my mother Ying Cao, my father Juwen Sun, for their endless love, understanding, and support throughout all these years. I would not be here if it not for you. Thank you for having faith in me during this challenging journey.

Contents

Abstract .... *i*

Acknowledgments .... *iv*

List of Tables .... *vii*

List of Figures .... *viii*

List of Abbreviations .... *x*

Chapter 1. Introduction 1

1.1 Research Background 1
1.2 Research Objectives 3
1.3 Our Approach and Research Contributions 3
1.4 Outline of This Dissertation 5

Chapter 2. Literature Review 7

2.1 Maximum Weighted Independent Set (MWIS) Problem 7
2.2 Process Planning and Scheduling (PPS) Problem 11
2.3 Summary: Observation and Our Solutions 16

Chapter 3. Maximum Weighted Independent Set (MWIS) Algorithms 17

3.1 Introduction 17
3.2 Definitions and Notations 18
3.3 MWIS Algorithms 19
3.4 Construction of the Algorithms 31
3.5 Reducing the Complexity of the Algorithm Using Approximation Algorithms 36
3.5.1 Discussion on the Complexity 36
3.5.2 Merging Approximation Algorithms with the Proposed MWIS Algorithm 37
3.6 Computational Experiment on MWIS Algorithms 42
3.7 Summary 50

Chapter 4. Formulation of the Resources Constrained Process Planning and Scheduling (PPS) Problem 52

4.1 Introduction 52
4.2 Process Planning and Scheduling Problem 54
4.2.1 Problem Description 54
4.2.2 Mathematical Formulation of the PPS Problem 57
4.2.3 Discussions on Formulating and Solving the PPS Problem via Conflicting Graph .. 60

4.3 Generating the Conflicting Graph 61
4.4 Weight Factors Calculation and the Configurations of MWIS Algorithms 73
4.4.1 The Weights Calculation 73
4.4.2 Weight Factor Arrangements with MWIS Algorithms 77
4.4.3 Heuristics Configurations 83
4.5 Solving the Example Problem via the Proposed Approach 84
4.6 Summary 92

Chapter 5. *Computational Experiments* 94

5.1 Integer Programming Model for Process Planning and Scheduling (PPS) Problem 94
5.1.1 Implementation of **I**nteger **P**rogramming (IP) Model 94
5.1.2 Numerical Results of IP Model 97
5.2 A Real-world Example Using the Proposed Approach 100
5.3 Results and Discussions on Test Instances 107
5.3.1 Scalability 108
5.3.2 Accuracy 111
5.4 Summary 116

Chapter 6. *Conclusions* 119

6.1 Algorithms for Maximum Weighted Independent Set (MWIS) Problem 119
6.1.1 Development of MWIS Algorithms 119
6.1.2 Performance of MWIS Algorithms 120
6.2 Approach for Process Planning and Scheduling (PPS) Problem 122
6.3 Research Contribution 123
6.4 Future Work 124
6.4.1 Improvements and Extensions 125
6.4.2 Integration with the sPLM System 126

Appendices 128

Appendix I: An Example for Algorithm A1 on a Simple Graph 128
Appendix II: Test Details of MWIS Algorithms 138
Appendix III: PPS Test Instances 139
Appendix IV: The PPS Test Results Summary on Accuracy 156
Appendix V: The PPS Test Results 159

References 212

VITA 224

List of Tables

Table 4-1. Operation Information of Part 1-4 54
Table 4-2. Unit Tasks and Nodes 76
Table 4-3. Heuristics Configurations 84
Table 4-4. Final Weight Factors Unit Task Candidates and Candidate Nodes via Heuristics #13 on the Example Problem 86
Table 4-5. Outputs of the Heuristics Configurations on the Example PPS Problem 92
Table 5-1. Integer Programming Model Numerical Results 98
Table 5-2. The Resource of a Job Shop – Machines and Tools 100
Table 5-3. The Technical Specifications for Part #1 102
Table 5-4. The Technical Specifications for Part #2 102
Table 5-5. The Technical Specifications for Part #3 103
Table 5-6. The Technical Specifications for Part #4 103
Table 5-7. Outputs of Heuristics on Real-world PPS Problem 106
Table 5-8. Outputs of Heuristics on Real-world PPS Problem without Tool Constraints 107
Table 6-1. Comparing the New Approach with Other Methods* 124

List of Figures

Figure 2-1. A Sample Interval Graph (Gardi, 2009) 10
Figure 2-2. Representation of the Process Plan adapted from Salehi and Bahreininejad (Salehi & Bahreininejad, 2011) 12
Figure 3-1. Three Types of Connected Unit Substructures (CUSs) 21
Figure 3-2. Compare Set at Level Node ‘3’ 27
Figure 3-3. Preliminary Set at Level Node ‘3’ 29
Figure 3-4. The Node Adding Procedures 32
Figure 3-5. A sample graph with 9 vertices 36
Figure 3-6. The CSS at Level Node ‘1’ 37
Figure 3-7. Merging Approximation Algorithms with the MWIS Algorithm Structure 41
Figure 3-8. Computation Time with Node Number of Algorithms A1 and A2 45
Figure 3-9. Computation Time with Edge Number of Algorithms A1 and A2 45
Figure 3-10. Computation Time with Node Number of Algorithms A3 and A6 46
Figure 3-11. Computation Time with Edge Number of Algorithms A3 and A6 46
Figure 3-12a. Computation Time with Node Number of Algorithms A4, A5, A7 and A8 47
Figure 3-13b. Computation Time with Node Number of Algorithms A4, A5, A7 and A8 (zoom-in) 47
Figure 3-14a. Computation Time with Edge Number of Algorithms A4, A5, A7 and A8 48
Figure 3-15b. Computation Time with Edge Number of Algorithms A4, A5, A7 and A8 (zoom-in) 48
Figure 3-16. The Average and Maximum Error Rate for All Algorithms 50
Figure 3-17. Performance of the MWIS Algorithms 51
Figure 4-1. Illustration of the PPS Example Problem 56
Figure 4-2. Reformatted Parts Information 62
Figure 4-3. Interpretation for Operation Data Preparation 63
Figure 4-4. Tasks Information with Simplified Duration Information 65
Figure 4-5. Transformed Tasks Information in Unit Tasks 66
Figure 4-6. Scheduling Problem Input Format 67
Figure 4-7. The Conflict Graph of operations $O2{,}1 \rightarrow O2{,}2$ of Part #2 and the operation $Oi, 1$ $(Ti, 1)$ of Part #i 70
Figure 4-8. Conflict Graph for the Example Problem 71
Figure 4-9. Graph Connection Details for the Example Problem 72
Figure 4-10. Updated Remaining Tasks Information for the Following Time Slot 87
Figure 4-11. Updated Remaining Conflicting Graph for the Following Time Slot 88
Figure 4-12. Updated Remaining Edge Connection Details for the Following Time Slot 89
Figure 4-13. Schedule Created with Heuristics #13 91
Figure 5-1. Parts Information with Simplified Duration Information 95
Figure 5-2. Input of Part #1 Operations 95
Figure 5-3. Inputs Dictionary Format for Package “pyomo” in Python 96
Figure 5-4. Computation Time with Changing ICI 99
Figure 5-5. Logarithmic Computation Time with Changing ICI 100
Figure 5-6. The description of 4 parts of PPS 101
Figure 5-7. Transformed Operations of Part #1 104
Figure 5-8. Schedule Created with Heuristics #19 105
Figure 5-9. Computation Time with Node Number of Heuristics #1~10 108
Figure 5-10. Computation Time with Edge Number of Heuristics #1~10 109
Figure 5-11. Computation Time with Node Number of Heuristics #11~28 110
Figure 5-12. Computation Time with Edge Number of Heuristics #11~28 111
Figure 5-13. The Average and Maximum Error Rate for All Heuristics Configurations 112
Figure 5-14. Details of Test Instances T24 114

Figure 5-15. The Average and Maximum Error Rate on Bad Test Instances 116
Figure 5-16. Performance of the Heuristics Configurations 117
Figure 6-1. Data Analytics with the sPLM System 127

List of Abbreviations

AMISL	All Maximal Independent Set Listing
AMIS	All Maximal Independent Set
CIM	Computer Integrated Manufacturing
CAD	Computer-Aided Design
CAM	Computer-Aided Manufacturing
CAPP	Computer-Aided Process Planning
CSS	Compare Set Subgraph
CUS	Connected Unit Substructure
GCP	Graph Coloring Problem
ICI	Input Complexity Index
IP	Integer Programming
MIS	Maximal Independent Sets
MWIS	Maximum Weighted Independent Set
PLM	Produce Lifecycle Management
PPS	Process Planning and Scheduling
PSS	Preliminary Set Subgraph
SD	Subgraphs Dictionary
SO	Stochastic Optimization
sPLM	Smart Product Lifecycle Management

Chapter 1. Introduction

*In this chapter, an overview of the research performed in this dissertation is presented. The chapter begins with an introduction of main topics of this research, (1) the **P**rocess **P**lanning and **S**cheduling (PPS) problem, and (2) the **M**aximum **W**eighted **I**ndependent **S**et (MWIS) problem. The research objectives and contributions are then addressed. Lastly, this chapter is wrapped up by outlining the structure of the overall dissertation.*

1.1 Research Background

Process **P**lanning and **S**cheduling (PPS) is to process a set of prismatic parts into completed products effectively and economically in a manufacturing system. A prismatic part to be produced is generally described by features. For each feature, one or more corresponding operations are determined according to its feature geometry and available machining resources. Each operation requires a selection of critical resources; some examples of these vital resources include machines, tools, fixtures, or specially qualified technicians. The resource constraints are that one critical resource cannot be occupied by more than one operation at the same time. There are precedence relationship constraints among operations, according to the geometrical and technological considerations. Process planning in PPS is the determination of an optimum process plan, i.e., operations and their sequences, within the precedence relationship constraints and resource constraints. The scheduling is the allocation of the resources in the machine shop over time to manufacture the various parts (Zhang et al., 2003). One of the common objectives is to find the feasible schedule with the earliest finishing time of all parts, or formally, minimizing the makespan. PPS as one of the main functions of **C**omputer-**A**ided **P**rocess **P**lanning (CAPP) system, it becomes more critical for the effective allocation and utilization of resources in modern flexible manufacturing systems. However, seeking an optimum integrated solution rapidly and effectively from all of the permutations, combinations of all of the tasks and resources according to specified criteria is challenging for the decision-makers (Zhang et al., 2014). Traditionally, such a problem is usually solved in a trial and error fashion using iterations, for instance, generic algorithms (Alander, 2014; Milosevic et al., 2016) and metaheuristics (Belfares et al., 2007; Bloechliger & Zufferey, 2013; Thevenin et al., 2018), or partially solved as an operation sequencing problem with individual part (Salehi & Bahreininejad, 2011; Su et al., 2018). However, such methodologies do not guarantee that an optimal solution is ever found, and they are usually slow and highly uncertain. In this research, we focus on a general type of the PPS problem with integrated resource allocation and process planning towards a typical objective, minimizing the makespan.

Without being restricted to the widely used methodologies, we would like to attack the PPS problem based on its nature. The nature of the PPS problem is to select a set of non-conflicting tasks that can be processed with available resources in parallel for each discrete time period. If tasks are represented as nodes, and the incompatibility between two tasks can be represented by an edge, then, the solution space of the PPS problem can be abstracted as the combinations of nodes in this conflicting graph. If a weight factor of each node can be introduced as the guideline for the node selection process, it is exactly solving the **M**aximum **W**eighted **I**ndependent **S**et (MWIS) problem.

The MWIS problem is one of the most important optimization problems in graph theory (Lovasz, 1994; Pardalos & Xue, 1994). It naturally arises in many applications, mainly in a scheduling environment. It considers a graph with weights assigned to nodes and seeks to discover the "heaviest" independent set, that is, a set of nodes with the maximum total weight so that no two nodes in the set are connected by an edge. The exact solution to the MWIS problem on general graphs is known to be NP-hard (Köhler & Mouatadid, 2016). Therefore, in order to utilize the concept of the MWIS in our PPS application, low-complexity algorithms for solving the MWIS problem that yields "good-quality" feasible solutions are desired.

1.2 Research Objectives

To overcome the drawbacks of the traditional methodologies for solving the PPS problem, the objective of this research is to **develop a new formulation of the PPS problem and solve it using the concept of the MWIS problem**. First, this new formulation shall integrate the two parts of the PPS problem, process planning and scheduling. Second, a direct mothed, which is solving the problem by a finite sequence of operations, is preferred for solving the PPS problem, and at the same time, ensure a reasonable accuracy. Third, the new formulation of the PPS problem shall be based on its nature, which is to select a set of non-conflicting tasks that can be processed with the available resources in parallel for each time period. Fourth, since the MWIS problem is a critical subproblem for solving the PPS problem by its nature, the "good-performance" MWIS algorithms are required. Lastly, the new approach for the PPS problem shall be tested and verified in terms of performance and feasibility.

1.3 Our Approach and Research Contributions

In this research, we propose a novel approach to formulate the PPS problem as a conflicting weighted graph. In such a graph, tasks and their resources selections are represented as nodes, the incompatibility between two tasks is represented by an edge. The solution space of the PPS problem is abstracted as the combinations of the nodes in such a conflicting graph. If the weight factor of each node is introduced to be the guideline for the node selection process, the process schedule with resource allocations is generated by solving the MWIS problem for each discrete time slot. Lastly, new MWIS algorithms are developed in order to solve the PPS problem efficiently.

The contributions of our approach are in the following areas:

Contributions on the MWIS problem: The MWIS algorithms are the determinants of the accuracy and computational speed in the proposed approach for the PPS problem. We propose a divide and conquer algorithm structure with relatively low complexity for solving the MWIS problem exactly. The proposed algorithm structure can also be used to improve the accuracy of existing low-complexity approximation MWIS algorithms. A set of "good-performance" MWIS algorithms are highlighted based on our PPS application. The detail of this contribution is presented in Chapter 3.

Contributions on the PPS problem: Unlike the commonly used iterative methods (such as generic algorithms and metaheuristics) or the mixed-integer programming approach, our approach provides a different angle to address the PPS problem and shows advantages over other approaches. The new approach requires minimum iteration. And it is guaranteed to return a feasible solution due to the nature of solving the MWIS problem on a conflicting graph. The new approach can be applied in a dynamic production environment, since the schedule of each time slot is computed separately. With carefully defined weight factors and "good-performance" MWIS algorithms, the new approach has satisfactory accuracy and computational speed. The detail of the proposed approach for the PPS problem is presented in Chapter 4, and the detail of computational experiments is presented in Chapter 5.

1.4 Outline of This Dissertation

The rest of this dissertation is organized as follows.

Chapter 2 provides a comprehensive literature review on the background, methodologies, and applications related to this work. Two major topics are reviewed in detail: (1) the MWIS problem and (2) the PPS problem. The findings, observations, and the proposed solutions based on the literature survey has been further analyzed to uncover the potential opportunities for the proposed new methodologies.

Chapter 3 discusses the development of new algorithms for the MWIS problem. These algorithms are the core functions for solving the resource-constrained PPS problem in later chapters. It starts with a quick introduction and the necessary graph theory background and definitions. Then, the proposed algorithms are explained in detail, and a detailed algorithm walkthrough is provided in Appendix I. Section 3.5 discusses merging the proposed MWIS algorithm with approximation MWIS algorithms to reduce the complexity. Then, Section 3.6 presents some illustrative numerical results to assess the performance of the algorithms in the context of the PPS application, and a set of “good-performance” algorithms are listed. Lastly, section 3.7 concludes the chapter.

Chapter 4 proposes a novel approach to formulate and solve the resource-constrained PPS problem via a conflicting graph. It starts with the introduction to the PPS problem. Then, the mathematical formulation of the PPS problem is presented. Section 4.3 discusses how the conflicting graph is generated, and Section 4.4 explains how to assign weight factors to the nodes in the conflicting graph. Then, section 4.5 takes an example from the literature to illustrate the proposed methodologies thoroughly. Lastly, section 4.6 concludes the chapter.

Chapter 5 presents the implementation and illustrative computational experiments of the integer programming model described in Chapter 4 as the baseline for further testing. Then, we verify the feasibility of the proposed approach for the PPS problem on a real-world example from literature. And further test results are reported and analyzed in terms of scalability, accuracy, and robustness. A set of satisfactory heuristics configurations are found based on the tests.

Chapter 6 concludes the dissertation and discusses the contributions of this research. Then, possible future directions for improving and extending this work are discussed.

Chapter 2. Literature Review

*In this chapter, a comprehensive literature review on the background, methodologies, and applications related to this work is carried out. Two major topics are reviewed in detail: (1) the **M**aximum **W**eighted **I**ndependent **S**et (MWIS) problem and (2) the **P**rocess **P**lanning and **S**cheduling (PPS) problem. As the conclusion of the review, the summary of findings, observations, and the proposed solutions based on the literature survey are presented at last.*

*2.1 **M**aximum **W**eighted **I**ndependent **S**et (MWIS) Problem*

As one of the most challenging problems in graph theory, the problem of finding the **M**aximum **W**eighted **I**ndependent **S**et (MWIS) can be stated as follows: for a graph where each node is assigned a weight, select a set of nodes, no two of which are adjacent, with the maximum possible total weight (Huang, 2013). We name such a graph as a conflicting weighted graph. The statement of the MWIS problem looks relatively simple; however, solving the MWIS problem on general graphs is computationally difficult. It has been shown to be an NP-hard problem (Köhler & Mouatadid, 2016), so it is unlikely to be solved in polynomial time.

One brute-force algorithm for exactly solving the MWIS problem amounts to checking all **M**aximal **I**ndependent **S**ets (MIS) and picking one with the maximum total weight. It follows that the MWIS problem is converted to the **A**ll **M**aximal **I**ndependent **S**ets (AMIS) listing (AMISL) problem (or maximal cliques listing problem in the complement graph). A pioneering work (Moon & Moser, 1965) has shown that any n-vertex graph has at most $3^{\frac{n}{3}}$ maximum cliques. Many algorithms are now known for the clique (or independent set) listing problem (Bron & Kerbosch, 1973; Loukakis & Tsouros, 1981; Johnson et al., 1988; Makino & Uno, 2004; Eppstein, 2005; Tomita et al., 2006; Cazals & Karande, 2008). Among those algorithms, a simple recursive backtracking algorithm (Bron & Kerbosch, 1973), Bron-Kerbosch algorithm named after its inventors, has been reported as the most successful clique listing algorithm in practice (Eppstein et al., 2010).

Other than the costly non-polynomial algorithm for the optimum solution on general graphs, people naturally go to three types of solutions: (i) solutions for special cases, it is known to be solvable in polynomial time in many cases including perfect graphs (Grotschel et al., 1993), interval graphs (Grotschel et al., 1993), disk graphs (Matsui, 1998), claw-free graphs (Minty, 1980), fork-free graphs (Alekseev, 2004), trees (Chen et al., 1988), sparse random graphs (Karp & Sipser, 1981; Czygrinow & Hanckowiak, 2006), circle graphs (Valiente, 2003), and growth-bounded graphs (Gfeller & Vicari, 2007). The MWIS problem has been found to be solvable in strongly polynomial time only on perfect graphs and their complements, on t-perfect graphs, and on claw-free graphs (Schrijver, 2003). (ii) approximation algorithms, there has been extensive work on approximating the MWIS (Halldorsson, 2004). The approximation can be achieved by using a greedy strategy (Furer & Kasiviswanathan, 2007). Sakai et al. (Sakai et al., 2003) investigated the performance guarantee of greedy algorithms to solve the MWIS problem. And (iii) there has been extensive work in the literature proposing a variety of heuristics (Kako et al., 2005). These specialized or heuristics algorithms have been developed for computing the exact MWIS (Fomin et al., 2006; Babel, 1994; Ostergard, 2002; Tassiulas & Ephremides, 1992) for limited types of graphs or graphs in general with certain trade-offs.

The Graph Coloring Problem (GCP) consists of assigning a single color (integer) to each vertex of an undirected graph, such that no two adjacent vertices share the same color, intending to minimize the number of colors (Tucker, 2012). The MWIS problem is a special case of the GCP, when each node is associated with a weight factor with an optimization objective of finding the set maximizing the total weight for each coloring terms of finding the optimum set of nodes for each color. The GCP, MWIS, and AMISL problems arise in many application domains, including resource allocation, scheduling, error-correcting coding, spatial statistics, and communication networks. Modeling scheduling problems as such problems are particularly relevant in the presence of incompatible entities to be scheduled, and multiple extensions of the GCP have been proposed to cope with these scheduling environments. We summarize the four scheduling problem formulations with GCP and its variations. Although these formulations are not fitting very well in the resource-constrained **P**rocess **P**lanning and **S**cheduling (PPS) problem considered in this dissertation, but they are inspiring for us to develop our approach.

(1) The Class/Exam Scheduling Problem

The class scheduling problem, also named as the timetabling problem, can be stated as follows: schedule a set of classes in a number of time slots such that no professor or student is required at the same time. Constraints can be mapped onto GCP as follows. Let each class be represented by a node. Attach two nodes by an edge if and only if there is a reason that the classes they represent may not be offered at the same time. Initially, there are two such reasons for nodes to be linked: either they are taught by the same instructor, or they are required by the same set of students. Upon adding in the links, color the graph. Each color represents a time slot available on a given timetable, so every node with the same color is offered at the same time. Similar applications of this problem can be the scheduling of classes and exams in a university, the scheduling of flights for an airline, and the scheduling of computing tasks to be run on a multiprocessor machine (Dandashi & Al-Mouhamed, 2010; Miner et al., 1995).

(2) The Interval Graph Scheduling

An interval graph is the intersection graph of a set of intervals of a real line, that is, a graph whose nodes correspond to intervals such that two nodes connected by an edge are associated with intersecting intervals, as shown in Figure 2-1 (Gardi, 2009). The intervals are representing the tasks, and the edges are indicating the incompatible tasks. The graph is then colored to find the mutual exclusion tasks that can be processed by the same resources. The interval graph scheduling and many variants of this problem have been extensively studied due to its numerous applications (Krarup & De Werra, 1982; Blazewicz et al., 2001; Zais & Laguna, 2016).

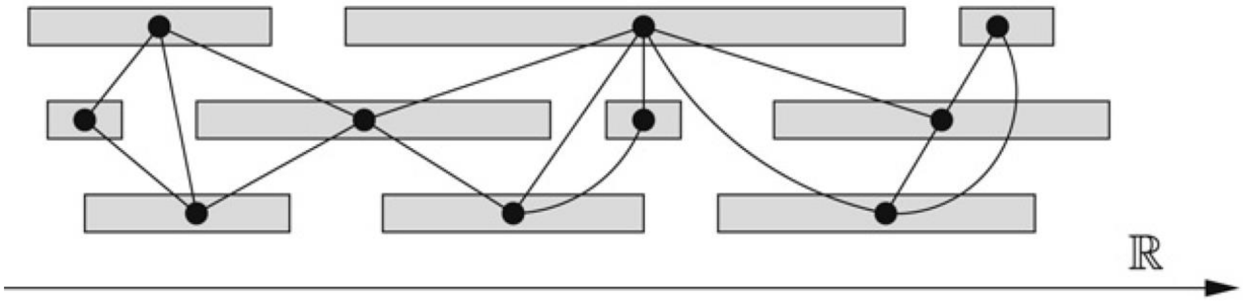


Figure 2-1. A Sample Interval Graph (Gardi, 2009)

(3) The Scheduling of Wireless Network

In a wireless network, two wireless nodes that transmit at the same resource (frequency), interfere with each other if they are located close-by. The scheduling problem is to decide which nodes should transmit at the given resource so that there is no interference, and nodes with longer queue length are given priority. If each node is given a weight equal to the queue length, it is optimum to schedule the set of nodes with the highest total weight. If a conflicting weighted graph is made, with an edge between each pair of interfering nodes, the scheduling problem is exactly the MWIS problem. This type of scheduling problem is mostly found in wireless communication applications (Tassiulas & Ephremides, 1992; Joo et al., 2013; Du & Zhang, 2016), but it is also applied in other types of applications (Duarte et al., 2015; Todosijevic & Mladenovic, 2016; Hansen et al., 2017; Gainanov et al., 2018).

(4) Graph Multi-coloring

The graph multi-coloring problem is an extension of the GCP. In this case, a node coloring corresponds to a sequence of colors (from the smallest to the largest). A node stands for a task, and an edge indicates that two tasks represented by the two end nodes of the edge are incompatible. Each color is a time slot, and each node must be assigned with a number of colors as defined by the processing time of the job. The objective is to minimize the number of used colors. Thevenin et al. apply this problem in a flow production environment (Thevenin et al., 2018). The graph multi-coloring problem formulation is the closest formulation comparable to our PPS problem. Still, it can only be applied in restricted conditions, such as each job requires the resources continuously and no subtasks of each job.

*2.2 **P**rocess Planning and **S**cheduling (PPS) Problem*

A job shop manufacturing environment is characterized by the make-to-order operation and the demands of small volumes with a large variety. Computer-aided process planning and scheduling systems have been developed to effectively support it. Computer-aided process planning (CAPP) is an essential interface between computer-aided design (CAD) and computer-aided manufacturing (CAM) in the computer integrated manufacturing (CIM) environment.

The resource-constrained **P**rocess **P**lanning and **S**cheduling (PPS) optimization problem can be defined as follows: Assuming there is a set of machining jobs in a machine shop, each job is referring to the production of a part. Each job consists of a set of machining operations (or tasks) to create features for the finishing part. These machining operations are processed in a sequence, which satisfies all the ordering constraints, and each operation requires specific combinations of critical resources. Some examples of these critical resources include machines, tools, fixtures, or special qualified technicians. One of the common objectives is to find a feasible schedule with the earliest finishing time of all jobs. In other words, this goal is to create a process plan with resource allocations minimizing the number of time slots needed to cover all operations.

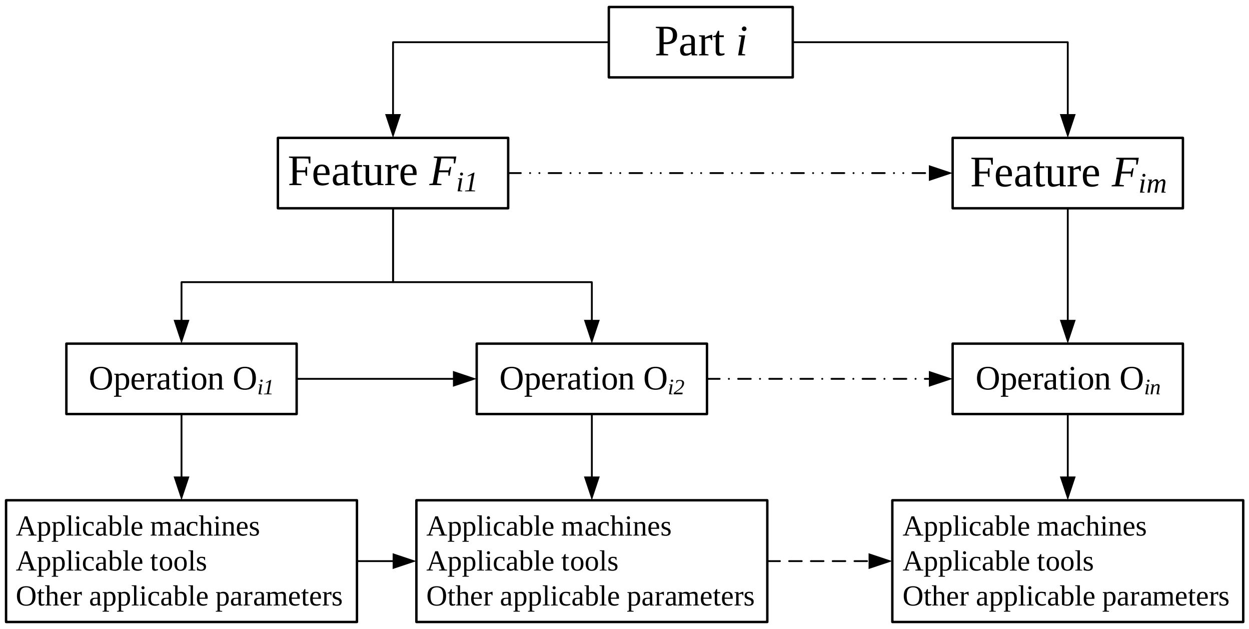


Figure 2-2. Representation of the Process Plan adapted from Salehi and Bahreininejad (Salehi & Bahreininejad, 2011)

Process planning and scheduling are usually complementary procedures. The former, process planning, can be used to plan manufacturing resources and operations for a part to ensure the application of good manufacturing practice and maintain the consistency of the desired functional specifications of the part during its production processes. Process planning activities include interpretation of design data, selection and sequencing of operations to manufacture the part, selection of machines and cutting tools, determination of cutting parameters, choice of jigs and fixtures, allocation of other resources required by the processes, and calculation of machining times and costs. To clarify process planning, parts are represented by manufacturing features. Figure 2-2 (Salehi & Bahreininejad, 2011) shows a part composed of m features in which each feature can be manufactured by one or more machining operations (n operations in total for the part). Each operation can be executed by several alternative plans if different machines, cutting tools, or set-up plans are chosen for this operation (Case & Harun Wan, 2000; Maropoulos & Baker, 2000). The latter, scheduling, specifies the schedule of manufacturing resources on each operation of the parts according to the importance of jobs, availability of resources and time constraints, and in the meantime, achieves the optimization objectives (Zhang et al., 2003).

PPS problems vary in complexity. However, seeking an optimum solution rapidly and effectively from all of the permutations, combinations of all of the tasks, manufacturing resources according to specified criteria is very difficult for decision-makers. Lenstra et al. (Lenstra et al., 1977) show that while some classical machine scheduling problems are efficiently solvable, others are NP-hard.

Due to its importance, practicality, and difficulty, in the past decade, many research studies have addressed the PPS problem. Traditionally, such a problem is usually solved in a trial and error fashion adopting methods such as generic algorithms and metaheuristics (Alander, 2014; Milosevic et al., 2016). These approaches include simulated annealing algorithm (Zhang et al., 2003; Tiwari et al., 2006; Li & McMahon, 2007; Chan et al., 2009), tabu search algorithm (Yan et al., 2003), agent-based approach (Shen et al., 2006; Wong et al., 2006), particle swarm optimization algorithm (Guo et al., 2006) and genetic algorithm (Zhang et al., 1997; Morad & Zalzala, 1999; Jia et al., 2002, 2003, 2007; Kim et al., 2003; Chan et al., 2005, 2006, 2008; Moon & Seo, 2005; Li et al., 2005; Zhang & Yan, 2005; Chan et al., 2006; Zhang & Gen, 2010; Salehi & Bahreininejad, 2011; Chaube et al., 2012; Qiao & Lv, 2012; Zhang et al., 2014). Researchers also solved the PPS problem partially as an operation sequencing problem with individual parts (Salehi & Bahreininejad, 2011; Su et al., 2018).

According to the discussions above, the integration and interactions of PPS are through an iterative and empirical fashion. The process planning system first generates a reasonable process plan for each part. Crucial processes in the system include determining suitable manufacturing resources (such as machines and tools), selecting set-up plans, and sequencing machining operations of the part. The scheduling system then specifies the schedule of manufacturing resources on each operation (task) of the parts according to the importance of operations, availability of resources, and time constraints. It is usually difficult to produce a satisfactory result in a single iteration of the execution of the two systems. For the process planning system, the decision of selecting machines and tools is usually made based on objectives to achieve the minimal manufacturing cost and ensure the good manufacturability of a part. Not all the generated process plans for a group of parts could be schedulable according to the time and resource feasibility in a job shop. To overcome this issue, it is necessary iteratively to re-invoke the process planning system to produce alternative plans for further trials until an acceptable scheduling solution is obtained. However, the above iterative process brings forth two severe problems in practical applications. First, it is quite tedious and time-consuming to search for a feasible solution to meet the requirements of process planning and scheduling simultaneously, and an overall optimized target is even more difficult to achieve. Meanwhile, the value of a process plan can be severely discounted since the assumption that all resources are available during the process planning stage might not be entirely valid in the scheduling stage. For instance, the generated process plans sometimes cause some machines to be overloaded, further, to create bottlenecks and restrict the capabilities of machines. Second, the PPS problem has vast solution spaces due to its combinatorial nature. Each time period can schedule one of the feasible operation sets, a feasible operation set can be any non-empty combination of feasible operations, and each operation can be one instance among all the feasible combinations of the available resources. The iteration-based approach needs to be carried out again and again in this vast discrete solution space. Furthermore, the outputs of such methodologies are easily trapped at local optimum, and the local optimum is hard to detect due to the combinatorial nature of such a problem.

Modeling a PPS problem as a GCP is particularly relevant in the presence of incompatible jobs. Multiple extensions of the GCP have been proposed to cope with these scheduling environments (Epstein et al., 2009; Fukunaga et al., 2012; Werra et al., 2005; Giaro et al., 2009; Halldórsson et al., 2004; Meuwly et al., 2010; Thevenin et al., 2018). As we identify in the previous section, the structural nature of some scheduling problems makes graph coloring an attractive formulation. Gamache et al. (Gamache et al., 2007) use graph coloring methods to determine a feasible schedule for crew scheduling problems within the airline industry. Moreover, they propose a new methodology to determine the existence of a feasible solution based on a graph coloring model and a Tabu search algorithm (Thevenin et al., 2018). However, these methodologies often require a specific application environment. For example, Blöchliger and Zufferey (Blöchliger & Zufferey, 2013), Thevenin et al. (Thevenin et al., 2018) formulate the PPS problem as a graph multi-coloring problem. They require that the production system uses continuous flow production, and each job is leading to the end product with no resource change. And still, unlike the particular case of the scheduling problem they are attempting, a typical PPS problem often requires multiple operations to be performed with different resource selections for each job following sequencing constraints. For those reasons, the graph multi-coloring formulations of the PPS problem could be limited in terms of universality.

2.3 Summary: Observation and Our Solutions

An in-depth review of the MWIS problem and PPS problem has been carried out in this Chapter. As a consequence, firstly, a closer integration of process planning and scheduling, is required. More specifically, determining the operation processing order in a machine shop and allocation of resources for each operation needs to be considered interactively. Secondly, a direct method or a method with fewer iterations is desired to solve the PPS problem.

Starting with the nature of the PPS problem, we proposed a novel approach to formulate a general type of the PPS problem with resource allocation and process planning integrated towards a typical objective, minimizing the makespan. The PPS problem is formulated into an undirected weighted conflicting graph. In this conflicting graph, nodes stand for operations and their resources; edges stand for constraints; weight factors are the guidelines for the node selection at each time slot. A variation of GCP, the MWIS problem, can be solved to find the best set of operations with their desired resources for each discrete time slot. This proposed approach can solve the problem directly, or it can be applied with few iterations for improving the quality of results.

The performance of the proposed approach depends on the accuracy and computational speed of the MWIS algorithms. We develop algorithms to compute the exact solution to the MWIS problem, and by utilizing the structure of the exact MWIS algorithms, we can improve the accuracy of existing MWIS approximation algorithms.

Chapter 3. <u>M</u>aximum <u>W</u>eighted <u>I</u>ndependent <u>S</u>et (MWIS) Algorithms

*In this chapter, we propose new algorithms for solving the **<u>M</u>**aximum **<u>W</u>**eighted **<u>I</u>**ndependent **<u>S</u>**et (MWIS) problem. These algorithms are the core functions for solving the resources constrained **<u>P</u>**rocess **<u>P</u>**lanning and **<u>S</u>**cheduling (PPS) problem in later chapters. Chapter 3 is organized in the following sections: Section 3.1 is the summary of the content of the chapter. Section 3.2 provides the necessary background and definitions of graph theory. Section 3.3 and Section 3.4 explain the proposed approach in detail, and Appendix I illustrates the proposed algorithm with a simple example. Section 3.5 discusses merging the proposed MWIS algorithm with approximation MWIS algorithms to reduce the complexity. Then, Section 3.6 presents some illustrative numerical results to assess the performance of the algorithms in the application context of the proposed approach for the PPS problem. Lastly, section 3.7 concludes the chapter.*

3.1 Introduction

The **<u>M</u>**aximum **<u>W</u>**eighted **<u>I</u>**ndependent **<u>S</u>**et (MWIS) problem considers a graph with weights assigned to nodes and seeks to identify the "heaviest" independent set, that is, a set of nodes with maximal total weight so that no two nodes in the set are connected by an edge. The MWIS problem arises in many application domains, including resource allocation, scheduling, error-correcting coding, spatial statistics, and communication networks. It has been shown to be combinatorial hard (NP-Hard) (Köhler & Mouatadid, 2016), and there has been extensive work in the literature proposing a variety of algorithms for solving the MWIS problem exactly or approximately. In this dissertation, we propose novel hybrid heuristic algorithms in a divide and conquer structure that yields optimum feasible solutions to the MWIS problem. We also solve the **<u>A</u>**ll **<u>M</u>**aximal **<u>I</u>**ndependent **<u>S</u>**ets (AMIS) listing (AMISL) problem, which can be seen as the subproblem of the MWIS problem in the same structure. Moreover, the proposed algorithm structure enables us to utilize available approximation algorithms (e.g., GWMIN and GWMIN2

(Sakai et al., 2003)) as subfunctions to get optimum or near optimum feasible solutions but much faster in computational speed. In the following chapters, we apply the proposed algorithms in the resources constrained **P**rocess **P**lanning and **S**cheduling (PPS) problem.

3.2 Definitions and Notations

Let G = (V, E) be a simple undirected graph with vertex set $V = \{1, \dots, v\}$, and a set of edges E. We denote by $|A|$ the cardinality of set A, so that the **edge number** of G is $|E|$ and the **node number** of G is $|V|$. Let $x \in V$, the **degree** (valence) of x is the number of edges with x as an endpoint. We denote the degree of x by $d_G(x)$. Let $Neig_G(x)$ denote the set of neighbors of vertex i and $Neig_G^+(x)$ denote $\{x\} \cup Neig_G(x)$. $d_G(x) = |Neig(x)|$ is the degree of vertex x.

In the graph G, let $S \subset V$ be any subset of vertices of G. Then, the induced subgraph $Ind_G(S)$ is the graph whose vertex set is S and whose edge set consists of all of the edges in E that have both endpoints in S (Diestel, 2006). For a vertex $k \in V$, let the complementary induced subgraph $C\_Ind_G(k)$ refers to the subgraph induced by all the node in V except node k, and the complementary neighbor induced subgraph $C\_Neig\_Ind_G$ (k) refers to the subgraph induced by the non-neighbors of k, and k is not in $C\_Neig\_Ind_G$ (k).

In the graph G, assume there is a sequence of vertices and edges x_0, e_1, x_1, e_2,…, e_n, x_n, where, for all $i = 1, \dots, n, x_{i-1}$ and x_i are the endpoints of e_i is called a **walk** (x_0, x_n-walk) in G from x_0 to x_n. A walk in which all edges are distinct is called a **trail** (x_0, x_n-trail) and a walk in which all vertices are edges are distinct is called a **path** (x_0, x_n-path). The length of this walk, trail, or path is n. The length of the shortest walk, trail, or path joining the vertex x to the vertex y is called the distance from x to y.

A connected, acyclic (no circuits) graph is called a tree. The components of an arbitrary acyclic graph are trees, and an acyclic graph is called a forest.

In the graph G, a subset $I \subseteq V$ is called an **independent set** (stable set, vertex packing) if the edge set of the subgraph induced by I is empty. An independent set is **maximal** (maximal independent set) if it is not a subset of any larger-size independent set, and **maximum** (maximum independent set) if there are no larger-size independent sets in the graph. The independence number $\alpha(G)$ (also called the stability number) is the cardinality of a maximum independent set in G. For each node $i \in V$, there is a positive weight $w_i > 0$. A subset of V can be represented by binary variable x_i, $(1 \le i \le |V|)$, where x_i is 1 if i is in the subset and 0 otherwise. A subset is called an independent set if no two nodes in the subset are connected by an edge. We are interested in finding the MWIS (Papadimitriou and Steiglitz, 1982), which can be expressed as an integer program:

$$\begin{aligned} &\max && \sum_i w_i x_i \\ &s.t. && x_k + x_i \le 1, \qquad (k,i) \in E \\ & && x_i \in \{0,1\}, \quad i \in V \end{aligned}$$

3.3 MWIS Algorithms

The proposed approach for the MWIS problem and AMISL problem has two phases following a divide and conquer structure: it starts by (a) removing nodes to get the induced subgraphs that are simple enough for finding the MWIS; and then by (b) iteratively adding nodes back one at a time, compare and merge to get the output. The first phase recursively partitions the graph into complementary induced subgraphs by removing nodes (and the adjunct edges) one at a time based on node removal heuristics. When induced subgraphs satisfy the desired patterns, these induced subgraphs become simple enough to be solved for MWIS with one comparison. A **Preliminary Set** (AMISL Preliminary Sets for the AMISL case) is found based on this complementary induced subgraph. The second phase of the algorithm adds back the nodes (and the adjunct edges) removed in the reversed sequence. At each adding, a **Compare Set** (AMISL sets for the AMISL case) is found to compare with the **Preliminary Set** (AMISL Preliminary Sets for the AMISL case). For the MWIS problem, the MWIS output set is the set with larger total weights among the Preliminary Set and the Compare Set of the current graph in the node adding process. For the AMISL problem, the AMISL output sets are the union of AMISL Compare Sets and AMISL Preliminary Sets for the graph with the adding node. The algorithm stops when all nodes (and the adjunct edges) are added back to the graph. With this brief understanding of the proposed approach, we are going into the details in the following sections.

3.3.1 Phase I: Dividing

Three types of unit graph structures (shown in Figure 3-1) are defined as **C**onnected **U**nit **S**ubstructures (CUS). The three types of CUS are: (a) an isolated node; (b) a pair of two connected nodes; and (c) a tree with a maximum diameter of 2 edges. Given an undirected weighted graph Γ consists of n different CUSs, CUS_1, CUS_2, …, CUS_i, …, CUS_n, $i \in \{1,2,\dots,n\}$, and no edge between these CUSs. Define $MWIS(\Gamma)$ as a set of nodes, and this set has the maximum total weight in Γ. We denote the $MWIS(\Gamma)$ as the MWIS of graph Γ, $MWIS(\mathrm{CUS}_1)$, $MWIS(\mathrm{CUS}_2)$, …, $MWIS(\mathrm{CUS}_\mathrm{i})$, …, $MWIS(\mathrm{CUS}_\mathrm{n})$ as the MWISs of the CUSs, respectively. The $AMIS(\Gamma)$ is a set of all maximal independent sets in Γ. We denote the $AMIS(\Gamma)$ as the AMIS of graph Γ, $AMIS(CUS_1)$, $AMIS(CUS_2)$, …, $AMIS(CUS_i)$, …,

$AMIS(CUS_n)$ as the AMIS of the CUSs, respectively. We denote the maximal independent set as $MIS_{CUS_i}^{k_i}$, which is an element in $AMIS(CUS_i) = \{MIS_{CUS_i}^{1}, MIS_{CUS_i}^{2}, \dots, MIS_{CUS_i}^{k_i}, \dots, MIS_{CUS_i}^{m_i}\}$, where $k_i \in \{1, 2, \dots, m_i\}$.

Theorem 3-1: For Base Cases in Recurrence
Given a graph Γ that consists of n different CUSs, $CUS_1, CUS_2, \dots, CUS_i, \dots, CUS_n$, $i \in \{1,2,\dots,n\}$, and no edge between these CUSs:

(i) For the MWIS problem, the $MWIS(\mathrm{CUS_i})$ can be found by one comparison in a CUS_i. The CUS with an isolated node can be considered as compared with an empty node set. For the Γ that consists of multiple CUSs, the $MWIS(\Gamma)$ is the union of the MWIS of each CUS in Γ, or formally,

$$MWIS(\Gamma) = MWIS(CUS_1) \cup MWIS(CUS_2) \cup \dots \cup MWIS(CUS_n)$$

(ii) For the AMISL problem, the $AMIS(CUS_i)$ can be found by dividing the graph into two independent node sets in a CUS_i. The CUS with an isolated node can be considered as dividing the graph into two node sets (one of the two sets can be an empty set, ϕ). For the Γ that consists of multiple CUSs, each CUS_i, $i \in \{1,2,\dots,n\}$, in Γ has its $AMIS(CUS_i)$. The $AMIS(\Gamma)$ of graph Γ is all the combinations of the MISs of all the CUSs, note that only picking one of the MISs from each CUS in one combination. For the $AMIS(\Gamma)$ of graph Γ, $k_i \in \{1, 2, \dots, m_i\}$, formally,

$$\begin{gathered} AMIS(\Gamma) = \\ \{ \\ \left\{MIS_{CUS_1}^{1}, MIS_{CUS_2}^{1}, \dots, MIS_{CUS_n}^{1}\right\}, \\ \dots, \\ \left\{MIS_{CUS_1}^{k_1}, \dots, MIS_{CUS_i}^{k_i}, \dots, MIS_{CUS_n}^{k_n}\right\}, \\ \dots, \\ \left\{MIS_{CUS_1}^{m_1}, \dots, MIS_{CUS_i}^{m_i}, \dots, MIS_{CUS_n}^{m_n}\right\} \\ \} \end{gathered}$$

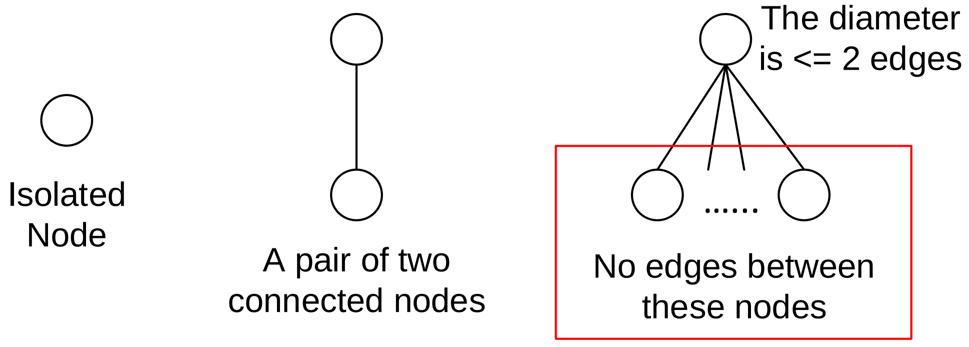

Isolated Node
A pair of two connected nodes
The diameter is <= 2 edges
......
No edges between these nodes

Figure 3-1. Three Types of Connected Unit Substructures (CUSs)

Proof of Theorem 3-1:

(i) For the MWIS problem, the MWIS can be found by one comparison in a CUS, because there are only two maximal independent sets in all the three types of CUSs. Since Γ consists of multiple CUSs, and there are no edges between these CUSs, the MWIS of each CUS does not have a conflict with the MWIS of another CUS in Γ. Because the MWIS of each CUS in Γ is the independent set with the possible maximum total weight, and MWISs of CUSs has no conflict with each other. We can get the union of MWISs of CUSs in Γ as the MWIS of Γ. ■

(ii) For the AMISL problem, the AMIS can be found by dividing a CUS into two independent node set. Since Γ consists of multiple CUSs, and there are no edges between these CUSs, the AMIS of each CUS does not have confliction with any node of another CUS in Γ. Because the AMISs of different CUSs in Γ has no confliction, get the union of the sets by choosing one set from the AMIS of each CUS and find all combinations without repeating of such unions. The union of each combination is one maximal independent set in the AMIS of Γ. ■

Corollary 3-1: The below statements in Theorem 3-1,
"For the Γ that consists of multiple CUSs,

(i) For the MWIS problem, the $MWIS(\Gamma)$ is the union of the MWIS of each CUS in Γ, or formally,
$$MWIS(\Gamma) = MWIS(CUS_1) \cup MWIS(CUS_2) \cup \ldots \cup MWIS(CUS_n)$$

(ii) For the AMISL problem, the $AMIS(\Gamma)$ of graph Γ is all the combinations of the MISs of all the CUSs, note that only picking one of the MISs from each CUS in one combination. For the $AMIS(\Gamma)$ of graph Γ, $k_i \in \{1, 2, \ldots, m_i\}$, formally,
$$\begin{gathered} AMIS(\Gamma) = \\ \{ \\ \left\{MIS_{CUS_1}^{1}, MIS_{CUS_2}^{1}, \ldots, MIS_{CUS_n}^{1}\right\}, \\ \ldots, \\ \left\{MIS_{CUS_1}^{k_1}, \ldots, MIS_{CUS_i}^{k_i}, \ldots, MIS_{CUS_n}^{k_n}\right\}, \\ \ldots, \\ \left\{MIS_{CUS_1}^{m_1}, \ldots, MIS_{CUS_i}^{m_i}, \ldots, MIS_{CUS_n}^{m_n}\right\} \\ \} \end{gathered}$$
",
also holds when the CUS is a general graph.

Proof of Corollary 3-1: In Corollary 3-1, the CUS in Γ in Theorem 3-1 is now a general graph. In other words, the connected components in Γ is a general graph. Similar to the proof of Theorem 3-1, because the MWISs or AMISs of these connected components in Γ has no conflict with nodes in a different connected component of Γ, so that Corollary 3-1 holds which means that Theorem 3-1 also holds when CUS is a general graph. ■

Theorem 3-1 and Corollary 3-1 show that we are able to find the MWIS and AMIS of an induced subgraph after partitioning it into a specific structure. In order to partition the graph to get an induced subgraph as Γ described in Theorem 3-1, we need to proceed in two steps: (a) break all the cycles in the graph, and (b) break the paths which are longer than 2 edges. In both steps, we need to remove the nodes (and the adjunct edges) which satisfying specific rules. We denote such a qualified node as a **removed node**.

Step 1: Break all cycles

First, we need to find a cycle basis of the given graph $G = (V, E)$. For each node $i \in V$ in G, count the number of basic (fundamental) cycles it belongs to, we denote the count for node i as $C_G(i)$. Then, we remove a node $n \in V$ to get the complementary induced subgraph $C\_Ind_G(n)$, where $C_G(n)$ is the maximum among all the $C_G(i)$. This process iterates until no cycle left in the induced subgraph. This induced subgraph left is either a tree or a forest, since all the cycles are broken by removing the node (and adjunct edges) belongs to the most cycles.

A basis for cycles of an undirected graph (**Cycle Basis**) is a minimal collection (a set of fundamental cycles) of cycles such that any cycle in the graph can be written as a sum of cycles in the Cycle Basis set (Diestel, 2012). Here summation of cycles is defined as “exclusive or” of the edges. The algorithm for finding a cycle basis is adapted from algorithm CACM 491, originally developed by K. Paton. For details on the algorithm and the production of the basic cycles, Paton’s original paper (Paton, 1969) should be consulted. Paton also discusses two other algorithms for basic cycle generation and contains performance statistics in the paper referred to. The adopted basic (fundamental) cycles algorithm can be depicted as in Algorithm 3-1 (Paton, 1969).

```
Algorithm 3-1: The Basic Cycles Algorithm
Input:
        A graph is finite, connected, undirected, and without loops or multiple edges.
Step 1:
        Let vertex 1 be the root of the spanning tree. Start forming the spanning tree by
        placing all edges of the form $\{1, W\}$ into the tree. At the same time, place all vertices
        W into a push-down list called STACK.
Step 2:
        Let Z be the last vertex added to STACK (i.e. the top of the stack). If STACK is
        empty, then stop. If STACK is not empty, then remove Z from STACK and go to step
        3.
Step 3:
        Consider all edges $\{Z, W\}$ which have not been examined. If all edges have been
        examined, go to step 2. Otherwise, for each edge $\{Z, W\}$ do the following:
            a. If W is in the tree generate the basic cycle formed by adding $\{Z, W\}$ to the
               tree and repeat step 3.
            b. If W is not in the tree, add $\{Z, W\}$ to the tree, W to STACK, and repeat
               step 3.
```

Algorithm 3-1: The Basic Cycles Algorithm (Paton, 1969)

Step 2: Break the paths which are longer than 2 edges to reduce the diameter of the components of the induced acyclic subgraph from step 1

If any of the connected components of the induced subgraph from step 1 has a diameter that is no less than 3 edges, remove the node in the middle of the longest path in that connected component of the graph. We name this node as the **Middle Node** of the path. For an odd path, the Middle Node is the midpoint of the path; for an even path, the Middle Node is one of the two nodes in the middle of the path. Algorithms 3-2 are adopted for checking the diameter, and Algorithms 3-3 is implemented for finding the Middle Node, respectively.

The diameter is the maximum eccentricity. The eccentricity of a node v is the maximum distance from v to all other nodes in G. If G is disconnected, the eccentricity of a node v is infinite. A diameter algorithm adapted based on the work by F.W. Takes, and his colleagues (Takes & Kosters, 2011; Takes & Kosters, 2013; Borassi et al., 2015) is applied here for computing the diameters in step 2. For each connected component of G, we utilize a function "`single_source_shortest_path_length`" from the python module "networkx" to compute the shortest path lengths from each node to all reachable nodes. The maximum value of the lengths found is the diameter of the connected component of G. We mark this algorithm as Algorithm 3-2, the diameter algorithm.

The Algorithm 3-3: the middle node algorithm is developed in order to find the middle point in a connected component of the induced acyclic subgraph. Since the input graph for finding the middle node is either a tree or a forest, we iteratively remove the nodes x (and the adjunct edges) whose degrees satisfy $d(x) = 1$ or $d(x) = 0$. The last one node removed is the middle node, if the path is odd. One of the last two nodes removed is one of the two middle nodes, if the path is even. This middle node algorithm is implemented as below.

Algorithm 3-3: The Middle Node Algorithm
Input:

The input graph, a tree or forest, is finite, undirected, and without loops or multiple edges. This input graph has at least ONE connected component whose diameter is greater than 2 edges.

Step 1:

Get a dictionary of the degrees of nodes in the input graph, namely "node_degree_dict", using the node name as keys and the degree value as values.

Step 2:

Find the keys which have values as 0 or 1, remove these nodes from the input graph to get the updated induced subgraph.

Step 3:

a. If the number of nodes in the updated induced subgraph is ZERO, the middle node is a node in the input graph (from step 1). Return this middle node.

b. If the number of nodes in the updated induced subgraph is not ZERO, clean the dictionary "node_degree_dict" and update the input graph with the updated induced subgraph. Then, start from step 1.

Algorithm 3-3: The Middle Node Algorithm

After the two steps of the node removal process, the induce subgraph satisfies the conditions as described in Theorem 3-1. We name the node $x \in V$ removed from G as a **removed node**. The complementary induced subgraph $C\_Ind_G(x)$ is called the induced subgraph at level node "x."

All the removed nodes and the associated components are stored in a dictionary named **Subgraphs Dictionary (SD)** with removed nodes as keys and the associated components as values for recording this process.

The number of removed nodes determines the number of iterations in both node removal and node adding processes so that we want to reduce the number of removed nodes to the greatest extend. By using the Algorithm 3-1, the basic cycles algorithm, we can break the cycles as many as possible at each removal so that we can reduce the graph to a tree with a minimum number of nodes removed. And by removing the middle node of the trees using the Algorithm 3-2, the diameter algorithm and Algorithm 3-3, the middle node algorithm, the diameter of the remaining trees are minimized, which is also minimizing the number of the node removed.

3.3.2 Phase II: Adding Nodes and Conquering

We consider a collection of problems that involve finding a feasible subset of the input of maximum weight. The input contains a collection of n distinguished elements, each carrying an associated nonnegative rational weight. Each set of distinguished elements uniquely induces a candidate for a solution, which we assume is efficiently computable from the set. The weight of a solution is the sum of the weights of the distinguished elements in the solution.

Halldorsson defines such a partitioning structure as the hereditary property (Halldorsson, 2000). A property is said to be hereditary if whenever a set S of distinguished element corresponds to a feasible solution, any subset of S also corresponds to a feasible solution. A property is semi-hereditary if under the same circumstances, any subset S' of S uniquely induces a feasible solution, possibly corresponding to a superset of S'. Theorem 3-2 is based on this partitioning idea.

Theorem 3-2: For Recurrence
For a given graph $G = (V, E)$, remove one node $n \in V$ (the removed node) to get the complementary induced subgraph $C\_Ind_G(n)$. Let $MWIS(G)$ denote the MWIS of graph G and let $AMIS(G)$ denote the AMIS of graph G.

(i) For the MWIS case, the $MWIS(G)$ is either the $MWIS[C\_Ind_G(n)]$ or the maximum weighted independent set that has node n as an element in graph G, $\{n\} \cup MWIS[C\_Neig\_Ind_G(n)]$. We name the $MWIS[C\_Ind_G(n)]$ as the **Preliminary Set** at level node n, and the $C\_Ind_G(n)$ as the **Preliminary Set Subgraph (PSS)** at level node n. Similarly, we name the set $\{n\} \cup MWIS[C\_Neig\_Ind_G(n)]$ as the **Compare Set** at level node n, and the $C\_Neig\_Ind_G(n)$ with node n as the **Compare Set Subgraph (CSS)** at level node n.

(ii) For the AMISL case, the AMIS of the complementary induced subgraph $C\_Ind_G(n)$ is formally $AMIS[C\_Ind_G(n)]$. All maximal independent sets which has node n as an element in each of the all maximal independent sets in graph G is formally $AMIS[Ind_G(n) \cup C_{Neig_{Ind_G}}(n) \cup \{n\}]$. The all maximal independent set of G, $AMIS(G)$, is the union of the $AMIS[C\_Ind_G(n)]$ and $AMIS[Ind_G(n) \cup C\_Neig\_Ind_G(n) \cup \{n\}]$. Note that if any maximal independent set in the AMISL outputs is a subset of another set in AMISL output sets in the union process. The subset is eliminated, since it is no longer a maximal independent set in the induced subgraph with node n. We name the $AMIS[C\_Ind_G(n)]$ as the **AMISL Preliminary Sets** at level node n. Similarly, we name the $AMIS[Ind_G(n) \cup C\_Neig\_Ind_G(n) \cup \{n\}]$ as the **AMISL Compare Sets** at level node n.

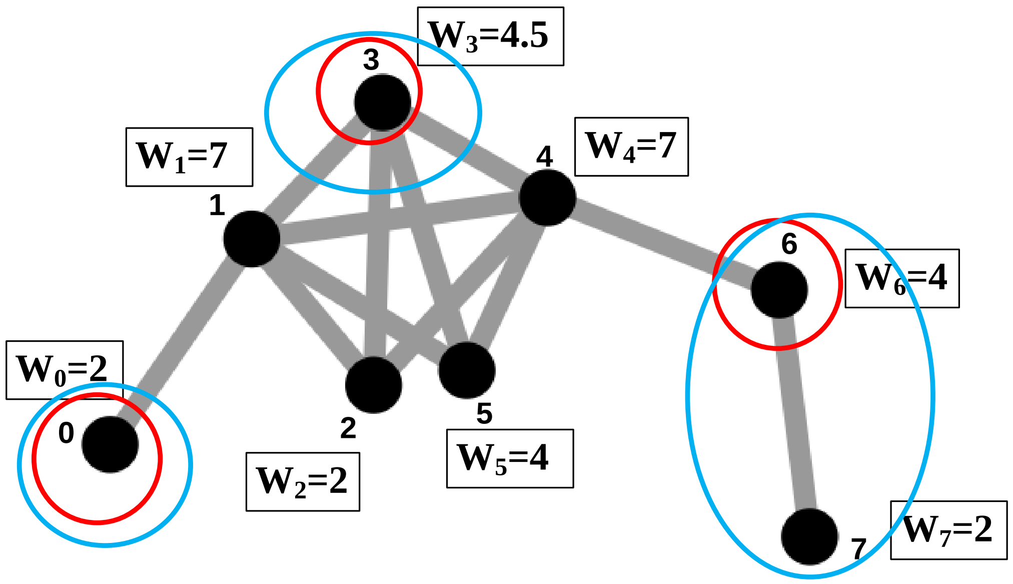


Figure 3-2. Compare Set at Level Node ‘3’

Let’s take an example to explain Theorem 3-2. Given a weighted graph G_{3-2} as Figure 3-2, the nodes, edges, node indexes, and weights associated is shown in the figure. Assuming node ‘3’ is the removed node, according to Theorem 3-2, the Compare Set at level node ‘3’ is the node set {‘0’, ‘3’, ‘6’} circled in red in Figure 3-2, and the Preliminary Set at level node ‘3’ is the node set {‘0’, ‘2’, ‘5’, ‘6’} circled in red in Figure 3-3. The $MWIS(G_{3-2})$ is either the set {‘0’, ‘3’, ‘6’} or {‘0’, ‘2’, ‘5’, ‘6’}. Since the set {‘0’, ‘2’, ‘5’, ‘6’} has a total weight 12 versus the total weight of {‘0’, ‘3’, ‘6’}, which is 11, the $MWIS(G_{3-2})$ is the set {‘0’, ‘2’, ‘5’, ‘6’}. In Figure 3-2, the induced subgraph in blue circles is the CSS, which is the $C\_Neig\_Ind_G(\{'3'\})$ plus node ‘3’. In Figure 3-3, the complementary induced subgraph, $C\_Ind_G(n)$ in the green circle is the PSS at level node n.

Proof of Theorem 3-2: by contradiction
(Since the MWIS and AMISL algorithms follow the same structure, we only prove the MWIS case here.) As the conditions described in Theorem 3-2, assuming all three statements always hold:

1. The Preliminary Set is the MWIS of $C\_Ind_G(n)$, $MWIS[C\_Ind_G(n)]$;
2. The Compare Set is $\{n\} \cup MWIS[C\_Neig\_Ind_G(n)]$;
3. There exists an Assumption Set in G. The Assumption Set is a maximal independent set that has a total weight greater than that of either the Preliminary Set or the Compare Set.

In the same graph G, since the Assumption Set, a maximal independent set in G, has a total weight greater than the total weight of the Compare set, and the Compare Set has the maximum possible total weight of the maximal independent set has node n as one element, the Assumption Set cannot contain node n as an element. Because the Preliminary Set has the maximum possible total weight of the maximal independent set in the complementary induced subgraph $C\_Ind_G(n)$, so that the maximum possible total weight of a maximal independent set without node n as an element is equal to the total weight of the Preliminary Set. Since the Assumption Set cannot contain node n as an element, then it must be a maximal independent set in the complementary induced subgraph $C\_Ind_G(n)$ and its total weight is no greater than the total weight of the Preliminary Set. It is a Contradiction with statement 3, which implies that such an Assumption Set does not exist. ■

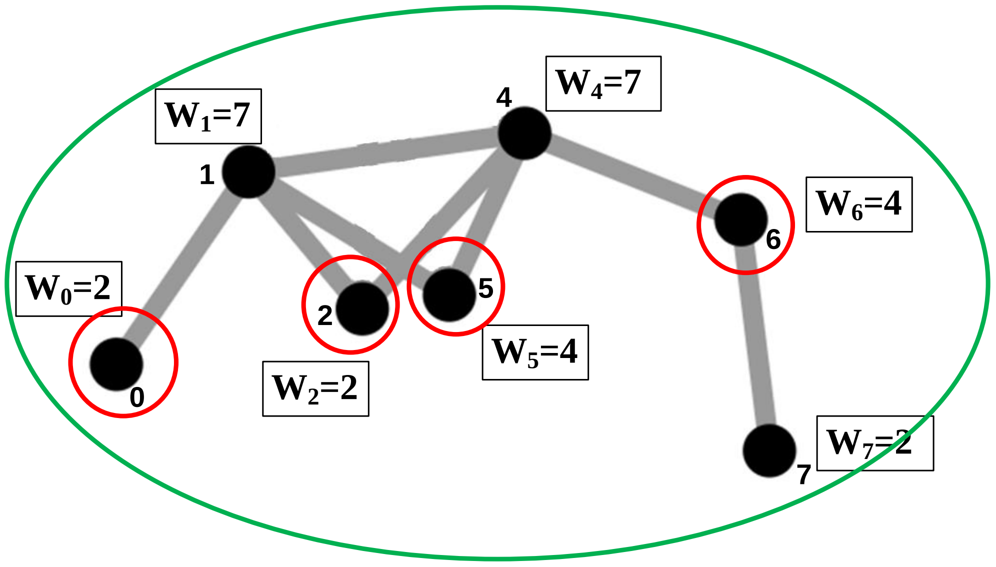


Figure 3-3. Preliminary Set at Level Node '3'

In order to further understand Theorem 3-2, suppose we decide to place a node v into a given maximum weighted independent set. It then suffices to search only in the non-neighborhood of v, $C\_Neig\_Ind_G(v)$, for the remaining nodes in the set. This suggests a natural heuristic, the greedy method. We can specify its result formally as

$$choose: v \in V$$

$$MWIS(G) \leftarrow \{'v'\} \cup MWIS[C\_Neig\_Ind_G(v)]$$

This rapid accumulation of an independent set by recursively looking at non-neighborhoods is attractive. Yet it remains disconcerting to completely ignore the neighborhoods of the pivot nodes, which may contain much larger weighted independent sets. Indeed, if we make a bad choice of a pivot node, we may be left with a minuscule set of independent vertices where there were plenty; thus, Greedy performs poorly in the worst case.

We are led to another rule for searching for an independent set. As before, choose a vertex and search in the non-neighborhood of that node. But this time also searches in the neighborhood of the pivot node, which makes the search area as $C\_Ind_G(v)$, and use whichever result has a heavier total weight. More formally,

$$choose\colon v \in V$$

$$MWIS\_AS(G) \leftarrow \max\left(\{'v'\} \cup MWIS\_AS[C\_Neig\_Ind_G(v)], MWIS\_AS[C\_Ind_G(v)]\right)$$

The discussions above are resulting Algorithm 3-4, MWIS algorithm structure (MWIS_AS), as below:

```
**MWIS_AS** ($G$), $G$ is a weight undirected graph.
**Begin**
    **If** $G = \emptyset$, **then return** [$\emptyset$]
    **Choose some** $v \in V$
    [$MWIS_1$]$\leftarrow$ $\boldsymbol{MWIS\_AS}[C\_Ind_G(v)]$
    [$MWIS_2$]$\leftarrow$ $\boldsymbol{MWIS\_AS}[C\_Neig\_Ind_G(v) \cup \{'v'\}]$
    **return (larger weight of** ($MWIS_1, MWIS_2$))
**End**
```

Algorithm 3-4: MWIS Algorithm Structure

AMISL algorithm follows the same structure, but we need to define a particular function called the Special Union. Assuming SS_1 and SS_2 are two sets of sets, the Special Union, $Spec\_\cup$ (SS_1, SS_2), which is a set, which is the union of all the sets in SS_1 and SS_2, and no set in $Spec\_\cup$ (SS_1, SS_2) is a subset of another set. This is resulting Algorithm 3-5, AMISL algorithm structure, (AMISL_AS) as below:

```
**AMISL_AS** ($G$), $G$ is a weight undirected graph.
**Begin**
    **If** $G = \emptyset$, **then return** [$\emptyset$]
    **Choose some** $v \in V$
    [$AMIS_1$]$\leftarrow$ $\boldsymbol{AMISL\_AS}[C\_Ind_G(v)]$
    [$AMIS_2$]$\leftarrow$ $\boldsymbol{AMISL\_AS}[C\_Neig\_Ind_G(v) \cup \{'v'\}]$
    **return** ($\boldsymbol{Spec\_\cup}(AMIS_1, AMIS_2)$)
**End**
```

Algorithm 3-5: AMISL Algorithm Structure

3.4 Construction of the Algorithms

From Theorem 3-1, we illustrate that the base cases for the divide and conquer algorithm structure. The base cases are constructed by removing nodes and the adjacent edges. We iteratively remove one node at a time by maximizing the number of cycles that the node belongs to in a cycle basis of the input graph or the current induced subgraph. Subgraphs dictionary (SD) is used to record this procedure. In SD, each node removed is the key and node sets of the connected components in the induced subgraphs as values of the keys, until the induced subgraphs satisfy the Theorem 3-1 conditions.

The node adding procedures that are illustrated in Figure 3-4, is based on Algorithm 3-4 and Algorithm 3-5. Assume there are m removed nodes for computing the MWIS or AMIS of graph G, the CSS and the PSS denote the Compare Set Subgraph and the Preliminary Set Subgraph, respectively. The MWIS algorithm or the AMISL algorithm needs to be executed on the CSS at level node l, $l \in \{1,2,\dots,l,\dots,m\}$, with n_l removed nodes to find the MWIS or the AMIS, respectively.

For the MWIS case, according to Theorem 3-2 and Algorithm 3-4, we can get the desired MWIS set by comparing the Compare Set and the Preliminary Set at each level of the removed node. The MWIS found at each level of the removed node is recorded in the subgraph MWIS dictionary (SMWISD): the current induced subgraph (the PSS plus the removed node at the level) is the key, and the MWIS found is the value. The SMWISD is used for searching the MWIS of the connected components, which is part of the Preliminary Set at the level.

For the AMISL case, according to Theorem 3-2 and Algorithm 3-5, we can get AMIS by comparing and merging the AMISL Compare Sets and the AMISL Preliminary Sets at each level of the removed node. The AMIS found at each level of the removed node is recorded in the subgraph AMIS dictionary (SAMISD): the current induced subgraph (the PSS plus the removed node at the level) is the key, and the AMIS found is the value. The SAMISD is used for searching the AMIS of the connected components, which is part of the AMISL Preliminary Set at the level.

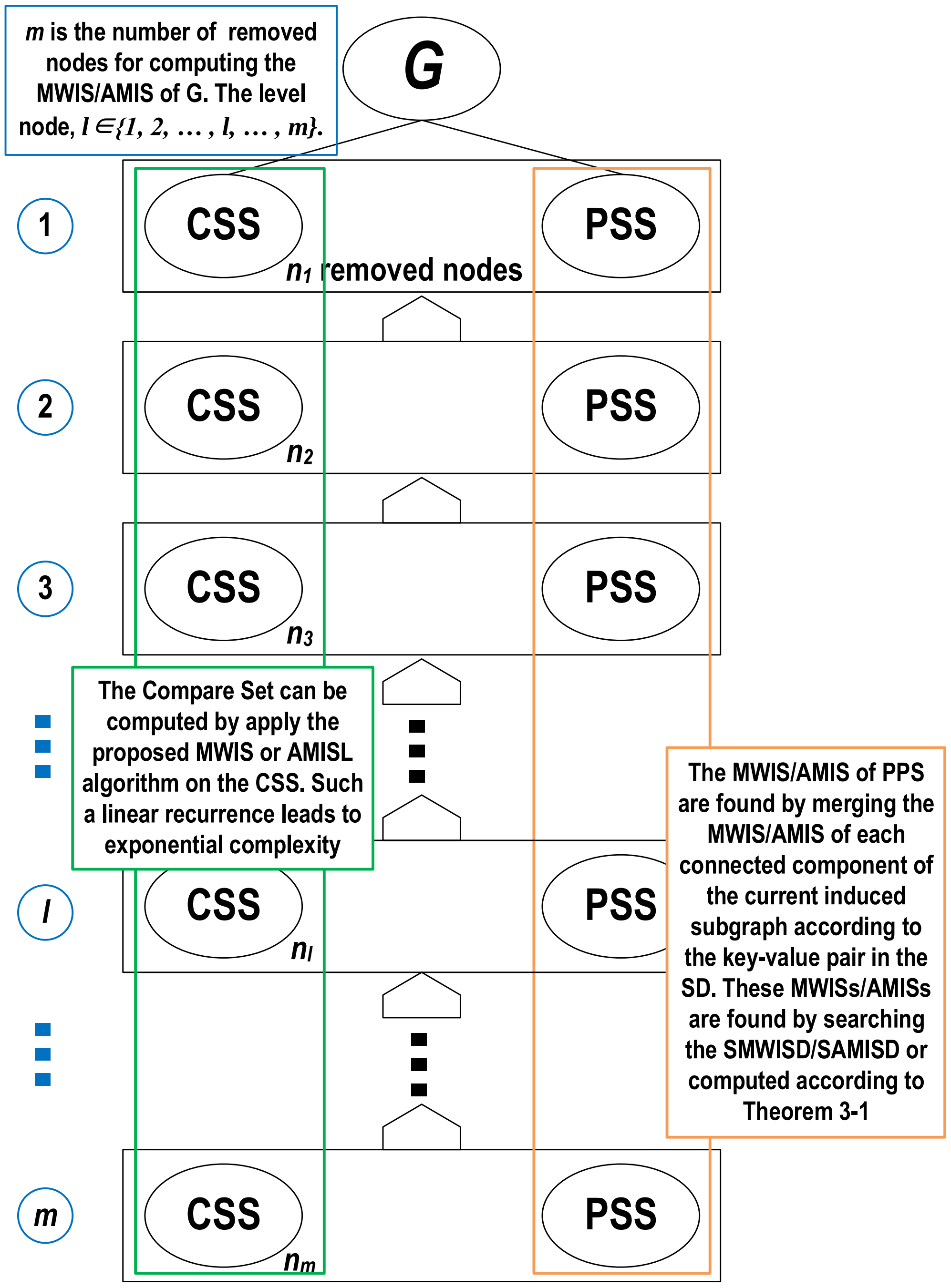

m is the number of removed nodes for computing the MWIS/AMIS of G. The level node, l ∈{1, 2, … , l, … , m}.
G
1
CSS
PSS
n1 removed nodes
2
CSS
PSS
n2
3
CSS
PSS
n3
The Compare Set can be computed by apply the proposed MWIS or AMISL algorithm on the CSS. Such a linear recurrence leads to exponential complexity
l
CSS
PSS
nl
The MWIS/AMIS of PPS are found by merging the MWIS/AMIS of each connected component of the current induced subgraph according to the key-value pair in the SD. These MWISs/AMISs are found by searching the SMWISD/SAMISD or computed according to Theorem 3-1
m
CSS
PSS
nm

Figure 3-4. The Node Adding Procedures

Together with Corollary 3-1, recurrence can be set up by adding the removed nodes back to the graph in the reverse order from the CUSs till getting the whole original graph. At each level of the removed node, the Preliminary Set and the AMISL Preliminary Set can be found as follows. For the MWIS case, we can get the Preliminary Set by aggregating the MWIS of each connected component of the current induced subgraph (without the removed node) according to the key-value pair in the SD. These MWISs are found by searching the SMWISD or computed according to Theorem 3-1. For the AMISL case, following the Theorem 3-1 and Corollary 3-1, we can merge the AMISs of all connected components of the current induced subgraph according to the key-value pair in the SD to get the AMISL Preliminary Set. These AMISs are found by searching the SAMISD or computed according to Theorem 3-1. While adding nodes back to get the Compare Set and the AMISL Compare Set, we follow the node adding heuristics for finding the Compare Set as below:

1. Get CSS, which is the induced subgraph by removing all neighbors of the removed node added; the removed node is included in the CSS.
2. Get the MWIS or AMIS of the CSS.
3. If the CSS getting from (1) does not satisfy the Theorem 3-1 conditions, perform the algorithm on this subgraph.

Thus, the Algorithm #1 MWIS (Algorithm A1) and Algorithm #2 AMISL (Algorithm A2) can be constructed as below:

```
Algorithm A1 MWIS: A hybrid heuristic algorithm for MWIS problem
Input: a weighted graph $G$
Output: MWIS of graph $G$.
Initializing: subgraphs dictionary (SD) = {}; subgraph MWIS dictionary (SMWISD) = {};
'last key' vertex = null.
Begin:
(1.1)   From step (1.1.1) to (1.1.5) Based on the input graph, find and remove the nodes one
        at a time, based on the node removal procedures, and update the SD: each node
        removed is the key and vertices sets of the connected components in the induced
        subgraphs as values of the keys, until the induced subgraphs satisfy the Theorem 3-1
        conditions.
(1.1.1) If the input graph satisfies the Theorem 3-1 conditions, go to step (1.2); if the input
        graph does not satisfy the Theorem 3-1 conditions, remove a vertex (the key in SD)
        and edges attached to it following the node removal steps in section 4.1, and get the
        component subgraphs vertices set(s) (value with the key);
(1.1.2) Update SD with the key-value pair;
(1.1.3) For each connected subgraph, exam whether it satisfies the Theorem 3-1 conditions;
(1.1.4) For those who do not satisfy Theorem 3-1 conditions, input these subgraphs to step
        (1.1.1); If the Theorem 3-1 conditions are satisfied, go to (1.1.5)
(1.1.5) When all subgraphs satisfy Theorem 3-1 conditions, return the latest SD and go to
        step (1.2).
(1.2)   Get the Preliminary Set by aggregating the MWIS of each connected component of
        the induce subgraph according to the last key-value pair in SD. These MWISs are
        found by searching the SMWISD or computed according to Theorem 3-1.
(1.3)   If 'last key' vertex = null, Compare Set is $\emptyset$; if not add the 'last key' vertex to the
        induced subgraph from (1.2) and follow the node adding heuristics to find the
        Compare Set at the level 'last key'.
(1.4)   Get the set with maximum total weight among the two sets: Preliminary Set and
        Compare Set at the level 'last key'. This set is the MWIS at the level 'last key' (the
        MWIS of the induced subgraph of the last level in SD). Update the SMWISD: the
        current induced subgraph from (1.3) is the key, and the MWIS found is the value.
(1.5)   Update SD by removing the last key-value pair. If the updated $SD = \{\}$, return the
        MWIS from step (1.4); if not, go to step (1.2).
```

Algorithm A1 MWIS: A Hybrid Heuristic Algorithm for MWIS Problem

For better describing the algorithms we proposed in this section, we provide a walkthrough of Algorithm A1 as well as all the terms in detail with a simple example in Appendix I. In the following section, we discuss the complexity of the proposed algorithms, and the means to improve the computational speed.

Algorithm A2 AMISL: A hybrid heuristic algorithm for AMISL problem
Input: a weighted graph G
Output: MWIS of graph G.
Initializing: subgraphs dictionary (SD) = {}; subgraph AMIS dictionary (SAMISD) = {}; ‘last key’ vertex = null.
Begin:

```
(2.1)   From step (2.1.1) to (2.1.5) Based on the input graph, find and remove the vertices
        one at a time, based on the vertices removal procedures, and update the SD: each
        vertex removed is the key and vertices sets of the connected components in the
        induced subgraphs as values of the keys, until the induced subgraphs satisfy the
        Theorem 3-1 conditions.
(2.1.1) If the input graph satisfies the Theorem 3-1 conditions, go to step (2.2); if the input
        graph does not satisfy the Theorem 3-1 conditions, remove a vertex (the key in SD)
        and edges attached to it following the node removal steps in section 4.1, and get the
        component subgraphs vertices set(s) (value with the key);
(2.1.2) Update SD with the key-value pair;
(2.1.3) For each connected subgraph, exam whether it satisfies the Theorem 3-1 conditions;
(2.1.4) For those who do not satisfy Theorem 3-1 conditions, input these subgraphs to step
        (2.1.1); If the Theorem 3-1 conditions are satisfied, go to (2.1.5)
(2.1.5) When all subgraphs satisfy Theorem 3-1 conditions, return the latest SD and go to
        step (2.2).
(2.2)   Following the Theorem 3-1 and Corollary 3-1, merge the AMISs of all connected
        components of the induce subgraph according to the last key-value pair in SD to get
        the AMISL Preliminary Set. These AMISs are found by searching the SAMISD or
        computed according to Theorem 3-1.
(2.3)   If ‘last key’ vertex = null, Compare Set is $\emptyset$; if not add the ‘last key’ node to the
        induced subgraph from (2.2) and follow the node adding heuristics to find AMISL
        Compare Sets at the level ‘last key’.
(2.4)   Get the Special Union of the two sets of sets: AMISL Preliminary Set and AMISL
        Compare Set at the level ‘last key’. Note that if any maximal independent set in the
        union is a subset of another set in this union process, eliminate this set from the union.
        This union is the AMISL output at the level ‘last key’ (the AMIS set of the induced
        subgraph of the last level in SD). Update the SAMISD: the current induced subgraph
        from (2.3) is the key, and the AMIS found is the value.
(2.5)   Update SD by removing the last key-value pair. If the updated $SD = \{\}$, return the
        AMIS from step (2.4); if not, go to step (2.2).
(2.6)   Find the MWIS based on the AMIS.
```

Algorithm A2 AMISL: A Hybrid Heuristic Algorithm for MWIS/AMISL Problem

3.5 Reducing the Complexity of the Algorithm Using Approximation Algorithms

3.5.1 Discussion on the Complexity

The runtime of the proposed Algorithm A1 and A2 highly depends on the input graph. In the Algorithm A1, the node adding procedures through step (1.2) to step (1.5), the Preliminary Sets are computed based on the CUS, or they may inherit the MWIS of previous induced subgraph before adding the node. By searching the dictionary, which stores the results of previous node adding steps, computations for Preliminary Sets are at low cost. But computations for Compare Sets may require executing Algorithm A1 on the CSSs according to the node adding heuristics. This leads to exponential complexity.

Let us take the graph G_{3-5} in Figure 3-5 as an example to illustrate the complexity of the proposed algorithm structure, to simplify the problem, assuming weights of the vertices are the same as the vertex index.

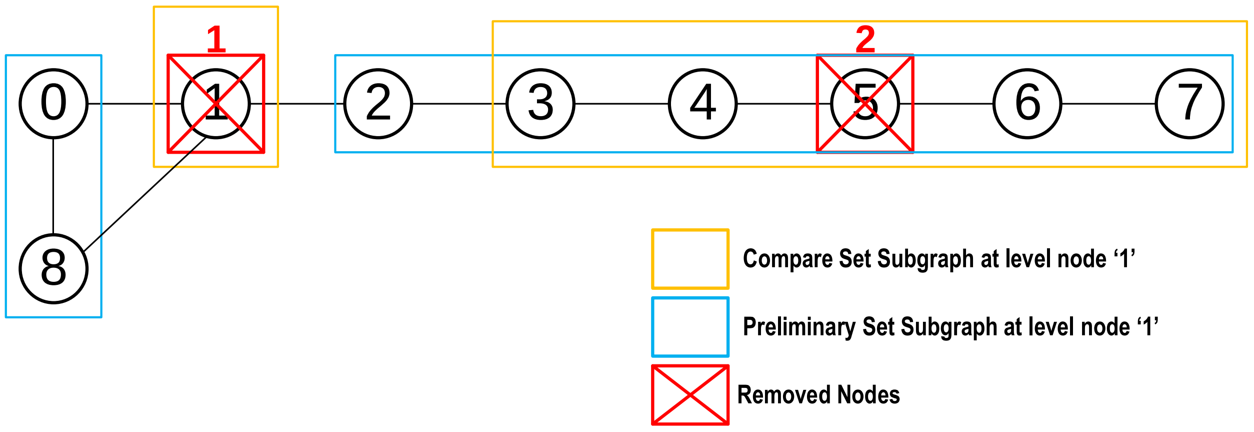

1
2
0
1
2
3
4
5
6
7
8
Compare Set Subgraph at level node '1'
Preliminary Set Subgraph at level node '1'
Removed Nodes

Figure 3-5. A sample graph with 9 vertices

Based on step (1.1) in Algorithm A1,

$$SD = \{'1': [\{'0','8'\},\{'2','3','4','5','6','7'\}],'5': [\{'2','3','4'\},\{'6','7'\}]\}$$

At level node ‘5’, the Preliminary Set is {'2','4','7'} and the Compare Set is {'3','5','7'} in the subgraph induced by nodes, $\{'5','2','3','4','6','7'\}$. The MWIS as level node ‘5’ is {'3','5','7'}. At level node ‘1’, based on the step (1.4), the Preliminary Set is the union the two MWIS of the two induced subgraphs (in the blue boxes), $Ind_{G_{3-5}}(\{'0','8'\})$ and $Ind_{G_{3-5}}(\{'2','3','4','5','6','7'\})$. The MWIS of $Ind_{G_{3-5}}(\{'0','8'\})$ is simple to know. The MWIS of $Ind_{G_{3-5}}(\{'2','3','4','5','6','7'\})$ is the same as the MWIS at level node ‘5’, which is {'3’, ‘5’, ‘7’}. But for the Compare Set, whenever the CSS does not satisfy the Theorem 3-2 conditions, we need to execute the Algorithm A1. Just like the CSS in the yellow boxes shown as Figure 3-6, it requires to execute Algorithm A1 to get the Compare Set at level node ‘1’, which is {'1’, ‘3’, ‘5’, ‘7’}. Such a linear recurrence leads to exponential complexity (Erickson, 2018). Note that, since Algorithm A2 follows a similar structure, but it is returning the AMIS at each step, the Algorithm A1 and A2 have the same complexity with the same input graph.

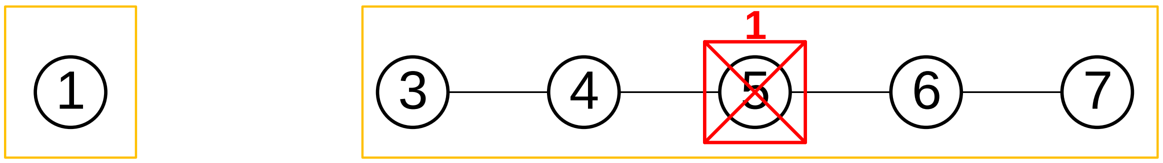


Figure 3-6. The CSS at Level Node ‘1’

3.5.2 Merging Approximation Algorithms with the Proposed MWIS Algorithm

Since calculations for the Compare Set slow down the execution of the proposed Algorithm A1 for the MWIS problem, we can speed up the computation by replacing Algorithm A1 on computing MWIS for Compare Sets with fast MWIS approximation algorithms. To illustrate this idea, we utilize two low complexity approximation algorithms to compute the Compare Set. Sakai et al. (Sakai et al., 2003) discuss greedy algorithms for the MWIS problem (GMIN-type algorithms). Two algorithms are the GMWIN and GMWIN2, which select a node of maximizing a node selection function, then remove it and its neighbors from the graph, and iterates this process on the remaining graph (induced subgraph) until no vertex remains. The set of selected nodes is the desired independent set. Let $G = (V, E, W)$ be a simple undirected graph with node set V, a set of edges E, and W is a set of weight factors associated with element in V. Let $u, v \in V$, for each $v_i \in V$ $(0 \leq i \leq |I| - 1)$, the two node-selecting functions are:

(1) GWMIN: maximizing $\frac{W_u}{d_{G_i}(u)+1}$

(2) GWMIN2: maximizing $\frac{W_u}{\sum_{u \in Neig^{+}_{G_i}(u)} W_u}$

Where, G_i is the remaining graph. We refer to the two simple greedy algorithms as Algorithm A3 GMWIN and Algorithm A6 GMWIN2, which are using the GWMIN and GWMIN2 node selection functions, respectively.

Let us consider the following framework of GMIN-type algorithms.

```
**Algorithm A3 GMWIN and Algorithm A6 GMWIN2,** GMIN-type Algorithm Framework
**INPUT**: A weighted graph G
**OUTPUT**: A maximal independent set in G
**begin**
    $I := \emptyset$; $i := 0$; $G_i := G$;
    **while** $V(G_i) \neq \emptyset$ **do**
        Choose a node based on a node-selecting function, say $v_i$, in $G_i$;
        $I := I \cup \{v_i\}$; $G_i + 1 := G_i[V(G_i) - Neig(v_i) + G_i(v_i)]$;
        $i := i + 1$;
    **od**
    Output $I$;
**end.**
```

Algorithm A3 and A6. The Algorithm GWMIN and Algorithm GMWIN2

As approximation algorithms, we are interested to know the lower bound of their accuracy. Sakai et al. (Sakai et al., 2003) proved the Theorem 3-3 and Theorem 3-4 as the lower bounds of the accuracy of the two algorithms.

Theorem 3-3. Algorithm A3 GWMIN outputs an independent set of weight at least $\sum_{v \in V} \frac{W_v}{d_G(v)+1}$.

Proof of Theorem 3-3:

$$\sum_{i=0}^{|I|-1} W_{v_i} \geq \sum_{i=0}^{|I|-1} \left(\sum_{u \in Neig^{+}_{G_i}(v_i)} \frac{W_u}{d_{G_i}(u)+1} \right)$$

$$\geq \sum_{i=0}^{|I|-1} \left(\sum_{u \in Neig^{+}_{G_i}(v_i)} \frac{W_u}{d_G(u)+1} \right)$$

$$= \sum_{v \in V} \frac{W_v}{d_G(v)+1} \ \blacksquare$$

Theorem 3-4. Algorithm A6 GWMIN2 outputs an independent set of weight at least $\sum_{v \in V} \frac{W_v^2}{\sum_{u \in Neig^{+}_{G}(v)} W_u}$.

Proof of Theorem 3-4:
Let $I = \{v_1, v_2, \dots, v_t\}$ be the independent set obtained by the algorithm. Let $f_G(v) = W_v / \sum_{u \in Neig^{+}_{G}(v)} W_u$.

$$\sum_{i=1}^{t} W_{v_i} \geq \sum_{i=1}^{t} \left(f_{G_i}(v_i) \times \sum_{u \in Neig^{+}_{G_i}(v_i)} W_u \right)$$

$$\geq \sum_{i=1}^{t} \left(\sum_{u \in Neig^{+}_{G_i}(v_i)} f_{G_i}(v_i) W_u \right) \quad \text{(from } f_{G_i}(v_i) \geq f_{G_i}(u) \forall u \in V(G_i)\text{)}$$

$$\geq \sum_{v \in V(G)} f_G(v) W_v \quad \text{(from } f_{G_i}(u) \geq f_G(u) \forall u \in V(G)\text{)}$$

$$= \sum_{v \in V} \frac{W_v^2}{\sum_{u \in Neig^{+}_{G}(v)} W_u} \ \blacksquare$$

With the approximation algorithms ready, we employ two different methods to merge an approximation algorithm with the proposed MWIS algorithm structure. Shown as Figure 3-7, in the step (1.3) of Algorithm A1, we denote the whole induced subgraph G_l at the level node 'l,' which is the PPS at the level node 'l' plus node 'l' (with the attached edges) in the node adding processes. Based on this assumption, the CSS at the level node 'l' is the induced subgraph of $C\_Neig\_Ind_{G_l}(l)$ plus the node 'l,' $C\_Neig\_Ind_{G_l}(l) \cup \{'l'\}$; the PPS is the complementary induced subgraph $C\_Ind_{G_l}(l)$. We can either apply an approximation algorithm on the whole induced subgraph G_l or the $C\_Neig\_Ind_{G_l}(l)$ for computing an MWIS as the Compare Set at the level node 'l.' Formally, for the two approximation algorithms, GWMIN and GWMIN2, four merged MWIS approximation algorithms are as follows:

(1) Algorithm A4 MWIS_CS_GWMIN: In the step (1.3) of Algorithm A1, when the CSSs do not satisfy the Theorem 3-1 conditions, instead of executing the Algorithm A1 on the CSSs, we compute Compare Sets based on the whole subgraph G_l using Algorithm A3 GWMIN.

(2) Algorithm A5 MWIS_SubCS_GWMIN: In the step (1.3) of Algorithm A1, when the CSSs do not satisfy the Theorem 3-1 conditions, we use Algorithm A3 GWMIN to compute MWISs on the $C\_Neig\_Ind_{G_l}(l)$, then plus node 'l' for Compare Set computations.

(3) Algorithm A7 MWIS_CS_GWMIN2: In the step (1.3) of Algorithm A1, when the CSSs do not satisfy the Theorem 3-1 conditions, we compute Compare Sets based on the whole subgraph G_l using Algorithm A6 GWMIN2.

(4) Algorithm A8 MWIS_SubCS_GWMIN2: In the step (1.3) of Algorithm A1, when the CSSs do not satisfy the Theorem 3-1 conditions, we use Algorithm A3 GWMIN2 to compute MWISs on the $C\_Neig\_Ind_{G_l}(l)$, then plus node 'l' for Compare Set computations.

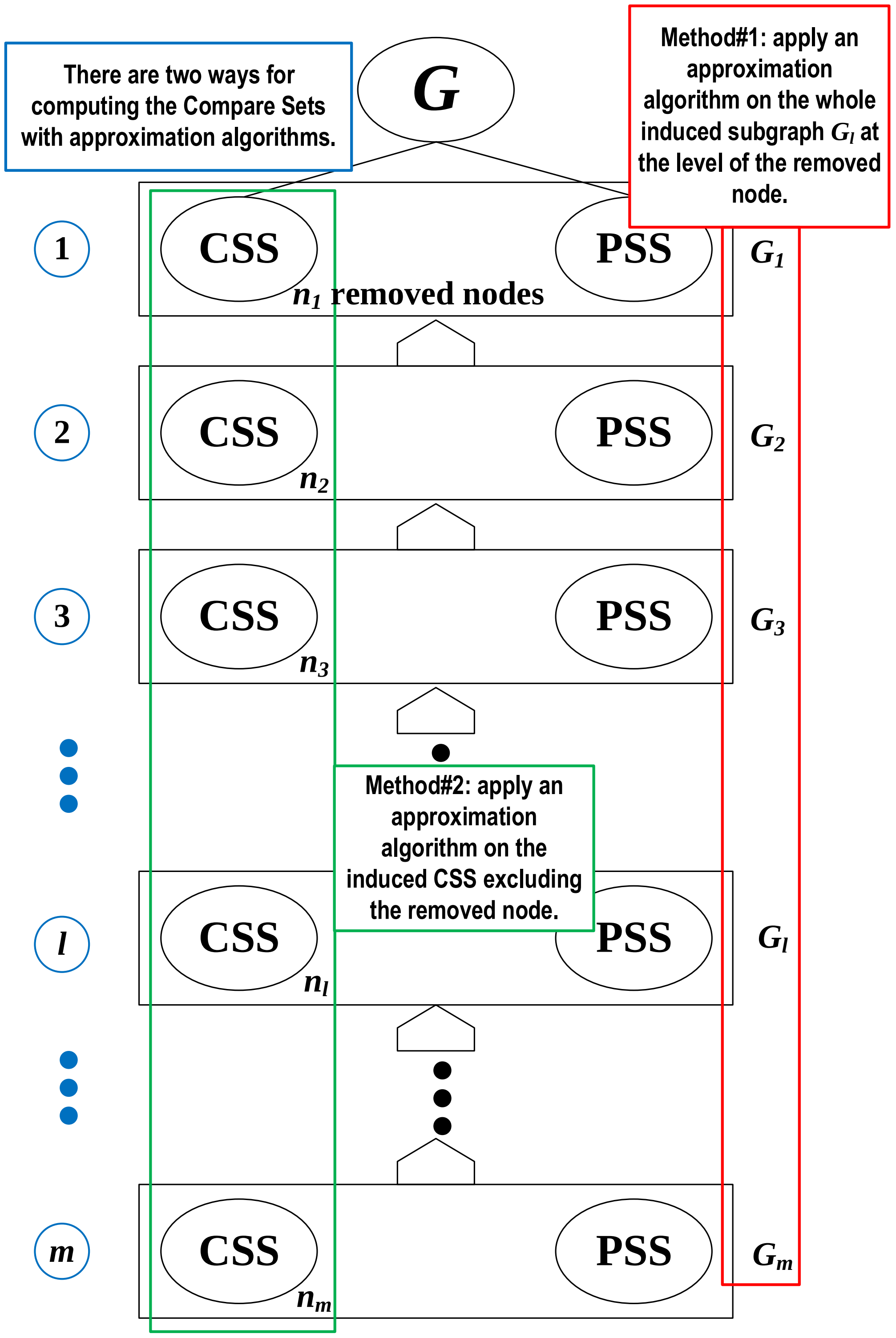

There are two ways for computing the Compare Sets with approximation algorithms.
G
Method#1: apply an approximation algorithm on the whole induced subgraph G_l at the level of the removed node.
1
CSS
PSS
n_1 removed nodes
G_1
2
CSS
PSS
n_2
G_2
3
CSS
PSS
n_3
G_3
Method#2: apply an approximation algorithm on the induced CSS excluding the removed node.
l
CSS
PSS
n_l
G_l
m
CSS
PSS
n_m
G_m

Figure 3-7. Merging Approximation Algorithms with the MWIS Algorithm Structure

According to Theorem 3-2, both composed MWIS approximation algorithms generate results no worse than the lower bound of the original approximation algorithms. In Algorithm A5 and Algorithm A8, the approximation algorithms are used on the $C\_Neig\_Ind_{G_l}(l)$, compare to Algorithm A4 and Algorithm A7, which are the approximation algorithms using the Algorithm A3 GWMIN and Algorithm A6 GWMIN2 on the whole subgraph G_l, respectively. By definition, the complementary neighbor induced subgraph, $C\_Neig\_Ind_{G_l}(l)$, is smaller than the whole induced subgraph G_l, because the node n and its neighbors are not included in $C\_Neig\_Ind_{G_l}(l)$. Theoretically, the Algorithm A5 and Algorithm A8 should have better accuracy than the Algorithm A4 and Algorithm A7, respectively. And the Algorithm A5 and Algorithm A8 should have a faster computational speed than the Algorithm A4 and Algorithm A7, respectively. The computational experiments in the following section also justify these conjectures.

3.6 Computational Experiment on MWIS Algorithms

According to the proposed approach for the Process Planning and Scheduling (PPS) problem discussed in Chapter 4, conflicting weighted graphs are created to test the scalability and accuracy of the algorithms in solving the PPS problem. Forty-three conflicting weighted graphs are created based on randomized PPS problems, from 5 nodes and 6 edges to 161 nodes and 4718 edges. The scalability analysis shows how the algorithms behave on the test graphs. It can be evaluated based on the computation time versus the different sizes of the test graphs, which measures by the node numbers and edge numbers of the different conflicting graphs. The accuracy refers to how likely the proposed approach can get to the optimum solution, MWIS. It can be measured by the average and the maximum error rate of all the test instances. The details of the results are shown in Appendix II.

Before we start the discussion on the scalability and accuracy, let us formally summarize all the MWIS algorithms to be tested as below:

- Algorithm A1 MWIS: the proposed exact MWIS algorithm.
- Algorithm A2 AMISL: the proposed exact AMISL-based MWIS algorithm.
- Algorithm A3 GWMIN: the GWMIN approximation algorithm from literature.
- Algorithm A4 MWIS_CS_GWMIN: it is an algorithm composed of Algorithm A1 and Algorithm A3. This algorithm computes Compare Sets based on the whole induced subgraph at each level using Algorithm A3 GWMIN.
- Algorithm A5 MWIS_SubCS_GWMIN: it is an algorithm composed of Algorithm A1 and Algorithm A3. This algorithm computes Compare Sets based on the induced CSSs, excluding the current removed node, using Algorithm A3 GWMIN.
- Algorithm A6 GWMIN2: the GWMIN2 approximation algorithm from literature.
- Algorithm A7 MWIS_CS_GWMIN2: it is an algorithm composed of Algorithm A1 and Algorithm A6. This algorithm computes Compare Sets based on the whole induced subgraph at each level using Algorithm A6 GWMIN2.
- Algorithm A8 MWIS_SubCS_GWMIN2: it is an algorithm composed of Algorithm A1 and Algorithm A6. This algorithm computes Compare Sets based on the induced CSSs, excluding the current removed node, using Algorithm A6 GWMIN2.

The computation time of Algorithms A1 and A2 changing with node number and edge number is shown in Figure 3-8 and Figure 3-9, respectively. Algorithms A1 and A2, as discussed in section 3.5, can be exponentially slow on certain graphs. The computation time can be hours when there are about 140 nodes and 4000 edges. Although the worst case of the two algorithms can be exponentially slow, the using scenarios of the PPS problem considered here may not always be the worst case. Algorithms A1 and A2 match higher-order (order 4 or higher) polynomial trendlines, but they are faster than the exponential trendline.

Figure 3-10 and Figure 3-11 show how the computation time changing with node number and edge number on Algorithms A3 and A6, respectively. Algorithms A3 and A6 are the approximation algorithms from literature, and they are the fastest among the 8 algorithms. The computation time is less than one second on the test graphs. Algorithms A3 and A6 are in lower-order polynomial complexity on the test graphs. The difference in the complexity of the two algorithms is due to the different greedy functions of the two algorithms.

Figure 3-12 and Figure 3-13 show how the computation time is changing with node number and edge number on Algorithms A4, A5, A7, and A8, respectively. Algorithms A4, A5, A7, and A8 are the composed algorithms based on Algorithm A1 structure with MWIS approximation algorithms. They are slower than the approximation algorithms utilized, but they are still much faster than the exact MWIS algorithms. The computation time is less than 45 seconds on the test graphs. Algorithm A5 and A8 are faster than Algorithm A4 and A7, respectively. This result of computational experiments matches the conjectures in section 3.5 that is the Compare Set computation is based on a smaller subgraph. And the Algorithm A7 and A8 are faster than Algorithm A4 and A5, respectively. This result also justifies that Algorithms A6 is faster than Algorithms A3 when the graph is relatively small (less than 3500 edges and less than 135 nodes.)

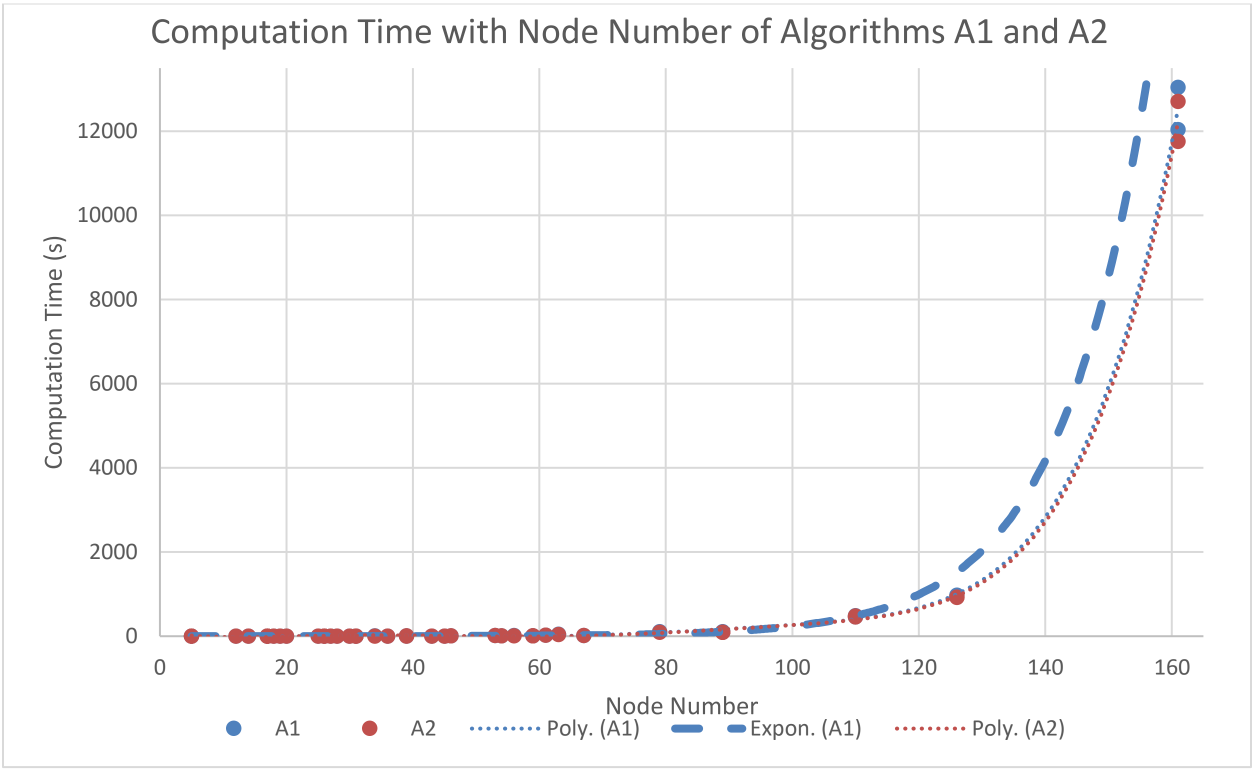


Figure 3-8. Computation Time with Node Number of Algorithms A1 and A2

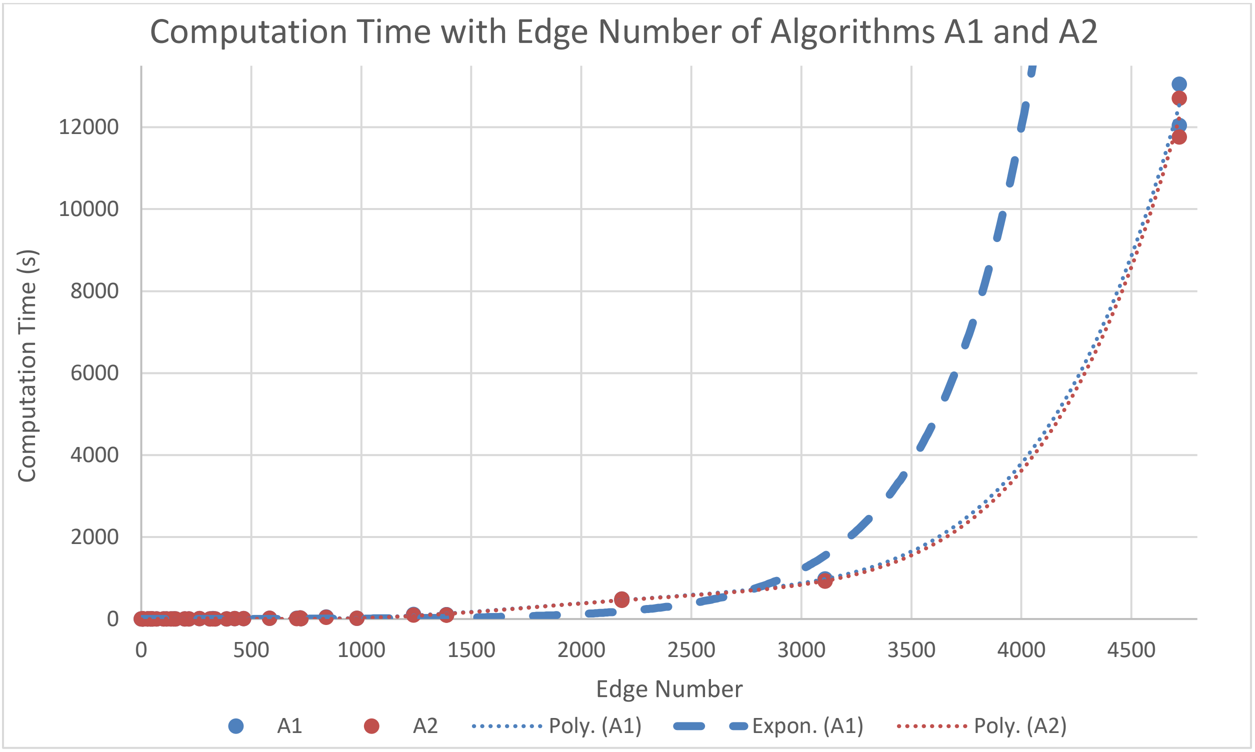


Figure 3-9. Computation Time with Edge Number of Algorithms A1 and A2

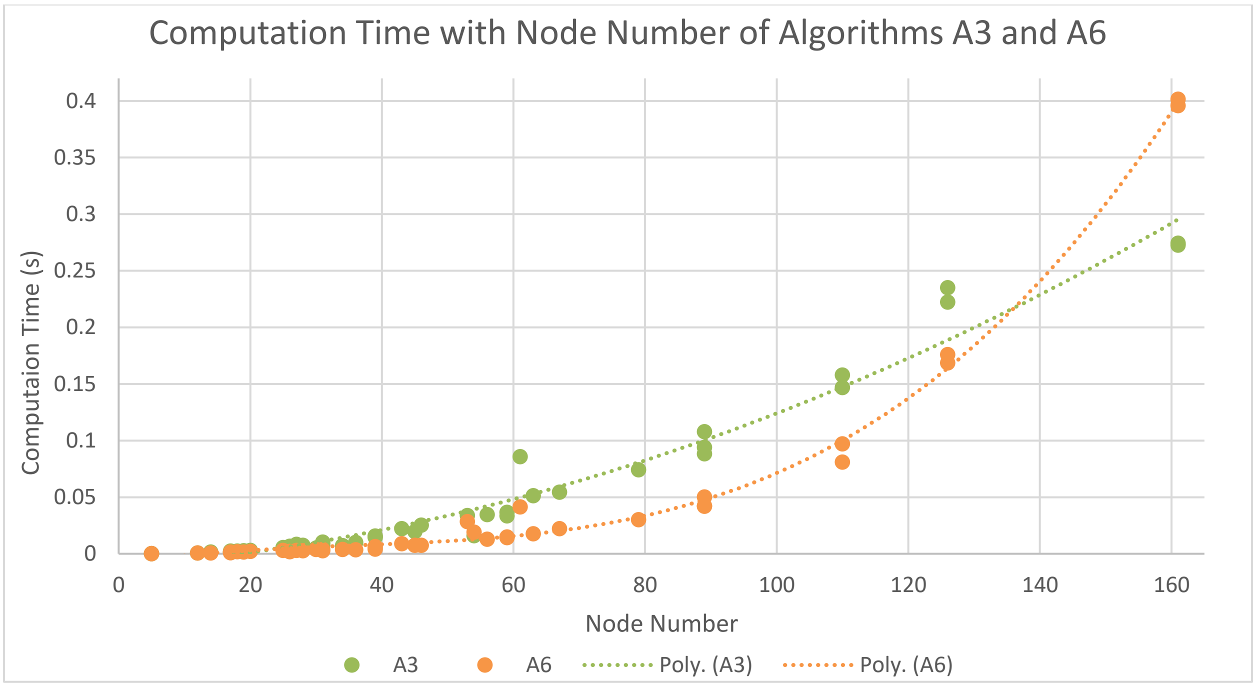


Figure 3-10. Computation Time with Node Number of Algorithms A3 and A6

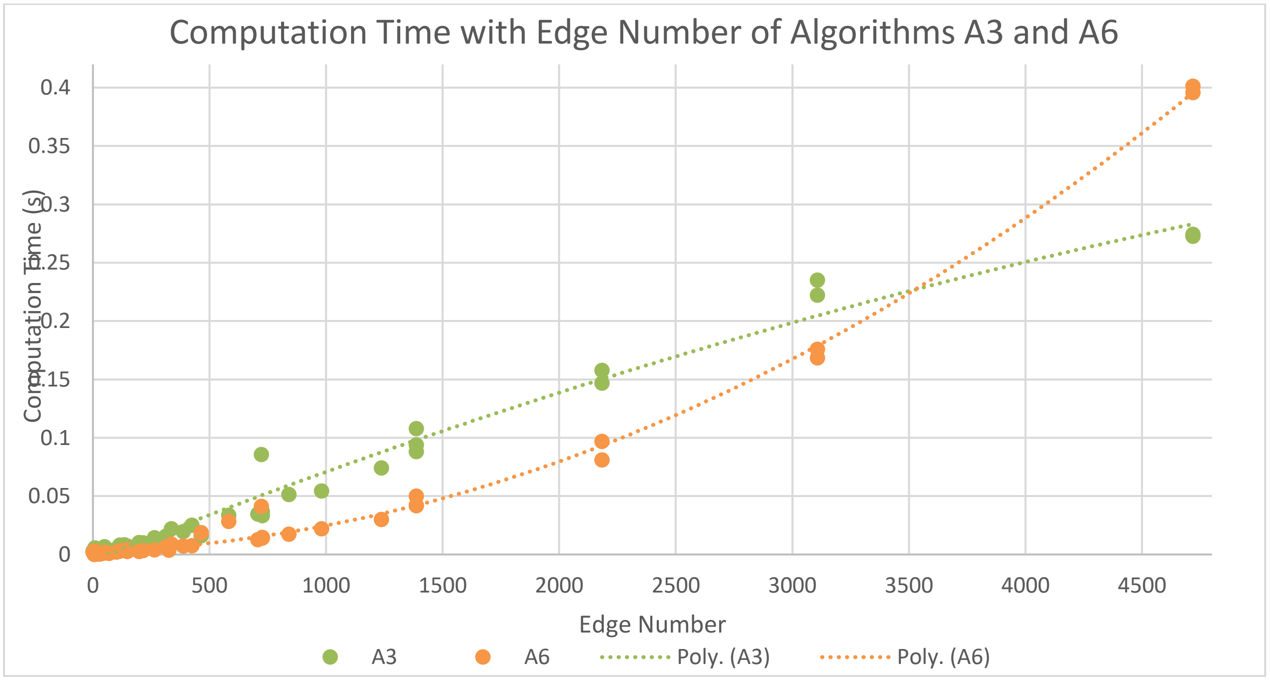


Figure 3-11. Computation Time with Edge Number of Algorithms A3 and A6

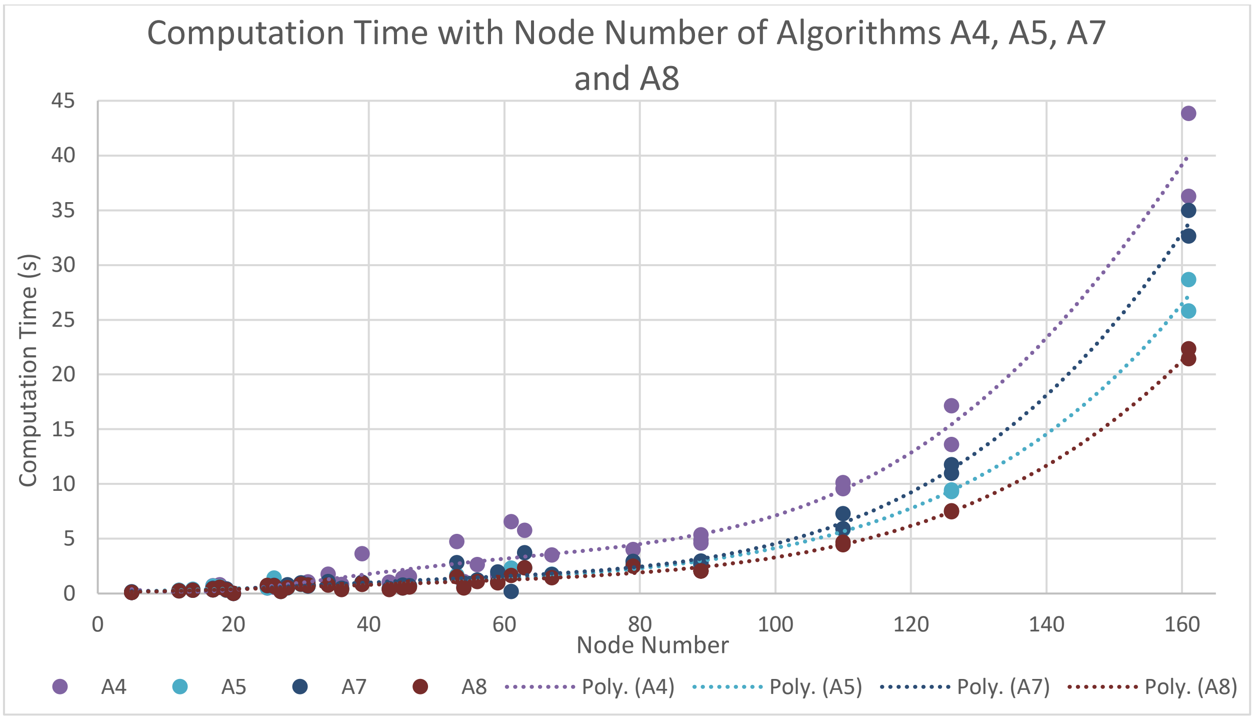


Figure 3-12a. Computation Time with Node Number of Algorithms A4, A5, A7 and A8

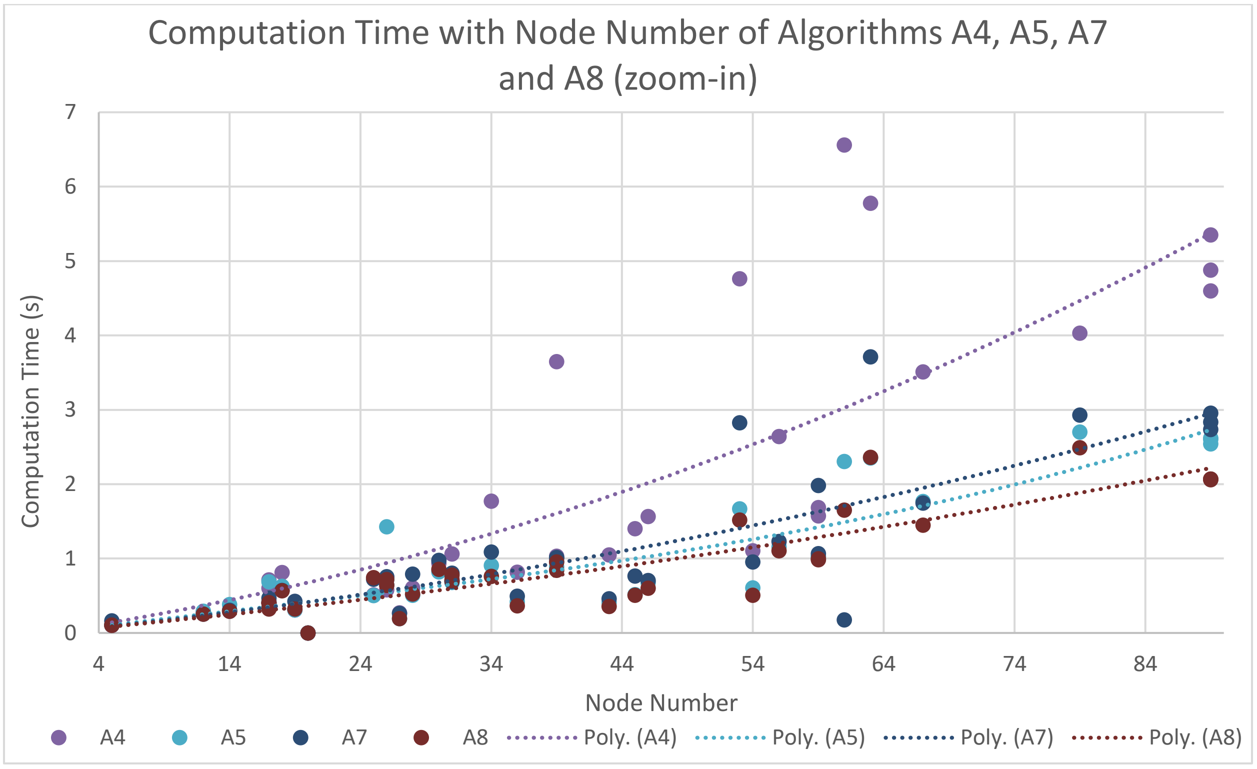


Figure 3-13b. Computation Time with Node Number of Algorithms A4, A5, A7 and A8 (zoom-in)

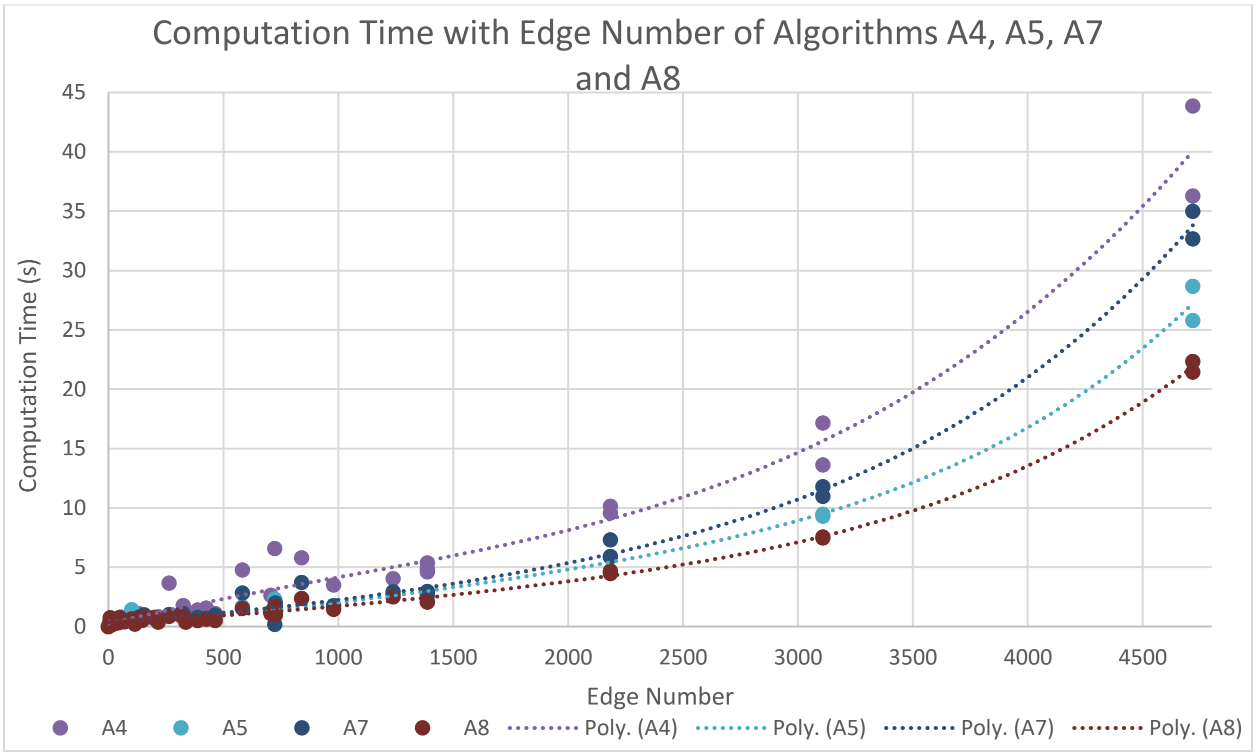


Figure 3-14a. Computation Time with Edge Number of Algorithms A4, A5, A7 and A8

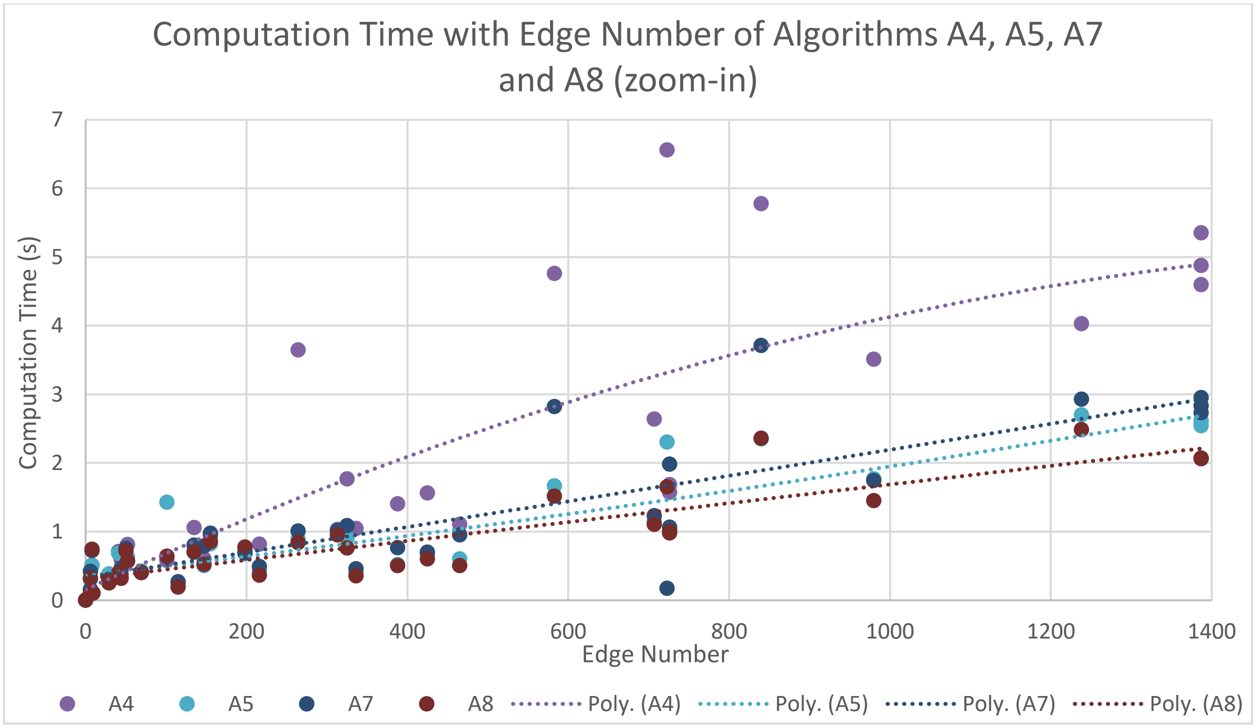


Figure 3-15b. Computation Time with Edge Number of Algorithms A4, A5, A7 and A8 (zoom-in)

Figure 3-14 shows the average and maximum error rate of the algorithms. Assume $W_{optimum}$ is the total weight of the optimum solution of the MWIS problem on the test graph, and W is the total weight of the MWIS set found by the algorithm. The weight error rate is calculated using the function below.

$$Weight\ Error\ Rate = \frac{W - W_{optimum}}{W_{optimum}} \times 100\%$$

Note that the Algorithms A1 and A2 shall return optimum solutions with the same total weight. And the test results justify this conjecture. This value is used as the baseline, $W_{optimum}$ for the weight error rate calculation.

The general accuracy of the algorithms can be listed below from the best to the worst:

1. Algorithm A1 MWIS
2. Algorithm A2 AMISL (same as Algorithm MWIS)
3. Algorithm A5 MWIS_SubCS_GWMIN
4. Algorithm A8 MWIS_SubCS_GWMIN2
5. Algorithm A4 MWIS_CS_GWMIN
6. Algorithm A3 GWMIN
7. Algorithm A7 MWIS_CS_GWMIN2
8. Algorithm A6 GWMIN2

As listed above, merging the approximation algorithms with Algorithm A1 structure can improve the accuracy. And the test results justify the statement that applying the approximation algorithm on smaller subgraphs can achieve better accuracy, e.g., Algorithm A5 and A8 have better accuracy than the Algorithm A4 and A7, respectively.

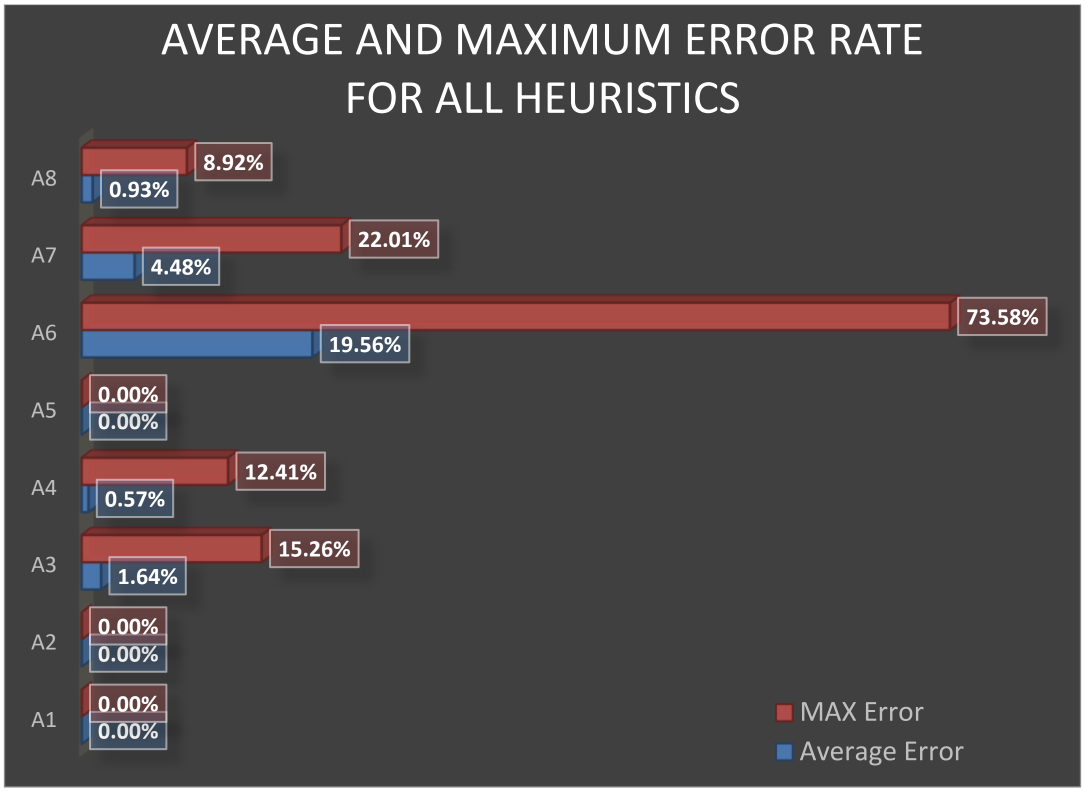


Figure 3-16. The Average and Maximum Error Rate for All Algorithms

3.7 Summary

In this chapter, we proposed new algorithms for exactly solving the MWIS problem. Moreover, based on the structure of the proposed MWIS algorithms, fast approximation algorithms GWMIN and GWMIN2 (Sakai et al., 2003) to are applied as a subfunction for finding sub-solutions on subgraphs. The merged algorithms are much faster than the original Algorithm A1 MWIS, and the accuracy of the outputs is no worse than the overall output of the approximation algorithm that is used as a subfunction. All the proposed algorithms and the approximation algorithms from the literature are tested on the conflicting graphs created based on PPS application scenarios. The overall performance of the algorithms is illustrated in Figure 3-17. The general accuracy of the best five algorithms can be listed below from the best to the worst:

Algorithm A1 MWIS; Algorithm A2 AMISL (same as Algorithm MWIS); Algorithm A5 MWIS_SubCS_GWMIN; Algorithm A8 MWIS_SubCS_GWMIN2; Algorithm A4 MWIS_CS_GWMIN. Note that all these algorithms considered satisfactory have the average error of less than 1% and the maximum error of less than 13% (The first four algorithms have the maximum error less than 9%) on all test instances.

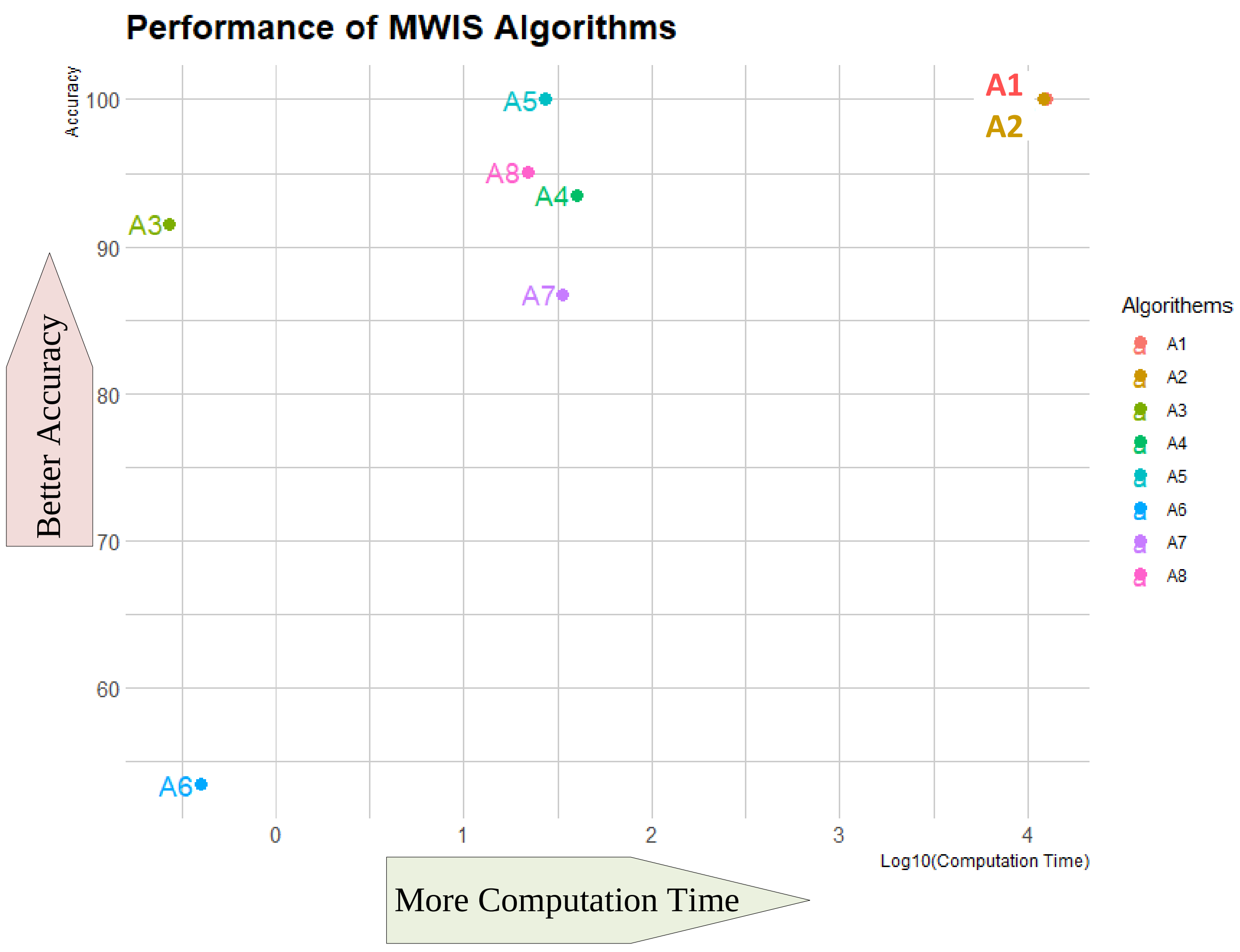


Figure 3-17. Performance of the MWIS Algorithms

We establish that we always obtain feasible solutions to the MWIS problem. And for all the general graphs we have tested, solutions to the exact MWIS algorithms are always optimum. In the following chapters, we apply the proposed algorithms for solving the resources constrained Process Planning and Scheduling (PPS) problem.

Chapter 4. Formulation of the Resources Constrained Process Planning and Scheduling (PPS) Problem

*In this chapter, we propose a novel approach to formulate and solve the resource-constrained Process Planning and Scheduling (PPS) optimization problem via a conflicting weighted graph. Using our approach, an optimized process schedule can be generated by solving the **M**aximum **W**eighted **I**ndependent **S**et (MWIS) problem using the proposed MWIS algorithms discussed in Chapter 3. Chapter 4 is organized as the following sections: Section 4.1 is the introduction to the PPS problem and the summary of the Chapter. Section 4.2 describes the PPS problem and formulates the mathematical model. Section 4.3 discusses how the conflicting graph is generated for the resource-constrained PPS problem. Section 4.4 explains how we configure the weight factors of the nodes in the conflicting graph with the proposed MWIS algorithms to achieve the optimization objective. Then, section 4.5 takes an example from the literature to illustrate the proposed methodologies thoroughly. Lastly, section 4.6 concludes the Chapter.*

4.1 Introduction

In this chapter, we propose a novel approach for formulating and solving the resource-constrained **P**rocess **P**lanning and **S**cheduling (PPS) optimization problem. The PPS problem can be defined as follows. Assuming there is a set of machining jobs in a machine shop, each job is referring to the production of a part. Each job consists of a set of machining operations (tasks) to create features for the finishing part. These machining operations are processed in a sequence, which satisfies all the ordering constraints, and each operation requires a particular combination of critical resources. Some examples of these critical resources include machines, tools, fixtures, or special qualified technicians. One of the common objectives is to find a feasible schedule with the earliest finishing time of all jobs. In other words, this goal is to create a process plan with resource allocations minimizing the number of time slots needed to cover all operations.

Based on the literature review in chapter 2, firstly, a closer integration of process planning and scheduling is required. More specifically, the determination of the operation processing order in a machine shop and the allocation of resources for each operation need to be considered interactively. Secondly, non-iteration or light-iteration methodologies with satisfactory accuracy are desired for the PPS problem. More specifically, the PPS problem is usually solved in a trial and error fashion using methods such as generic algorithms and metaheuristics. However, such methodologies do not guarantee an optimal solution is ever found, and they usually do not scale well with complexity. Also, these methods operating on dynamic data sets is difficult, as genomes begin to converge early on towards solutions which may no longer be valid for later data.

In our approach, the two procedures, the resource selection and process scheduling, in the PPS problem are integrated and formulated into an undirected weighted conflicting graph due to the nature of sequencing and resource constraints. A node in the conflicting graph represents one operation with one possible combination of its required resources during one time slot, and an edge indicates that there is a conflict between the two nodes at both ends of the edge. Each node in the graph is assigned with a weight factor as the guidance for the node selection process to fulfill the optimization objective. The node with a higher possibility leading to the objective, is given a larger weight, so that they are more likely to be selected when generating the schedule. We utilize algorithms proposed in Chapter 3 to solve the Maximum Weighted Independent Set (MWIS) problem to realize this node selection process to get the optimum or a near-optimum solution. A simplified PPS example problem from the literature (Zhang et al., 2014) is employed to illustrate the proposed approach.

4.2 Process Planning and Scheduling Problem

4.2.1 Problem Description

As an example of the PPS problem in a manufacturing system, there are four parts to be processed by four machines with a number of tools. Each part requires several operations (four parts have 4, 3, 3, and 4 operations, respectively), and each operation can be performed on at least one available machine with different processing times. Table 4-1 shows the operation information of the four parts. Each column describes the part ID, operation ID, successors, operation name, machine candidates, tool candidates, and machining time, respectively. The illustration of one feasible solution to this example problem is shown in Figure 4-2.

Table 4-1. Operation Information of Part 1-4

Part-ID	**Op-ID**	**Successor**	**Operations**	**Machine Candidates**	**Tool Candidates**	**Machining time (time unit)**
Part 1	$O_{1,1}$	$O_{1,2}$, $O_{1,3}$	Milling	M_2, M_3, M_4	T_6, T_7	40, 40, 30
	$O_{1,2}$	$O_{1,4}$	Milling	M_2, M_3, M_4	T_6, T_7	40, 40, 30
	$O_{1,3}$	$O_{1,4}$	Milling	M_2, M_3, M_4	T_6, T_7	20, 20, 15
	$O_{1,4}$	-	Drilling	M_1, M_2, M_3, M_4	T_2	12, 10, 10, 7.5
Part 2	$O_{2,1}$	$O_{2,2}$, $O_{2,3}$	Drilling	M_1, M_2, M_3, M_4	T_1	12, 10, 10, 7.5
	$O_{2,2}$	-	Milling	M_2, M_3, M_4	T_{12}	20, 20, 15
	$O_{2,3}$	-	Milling	M_2, M_3, M_4	T_6, T_7, T_{11}	18, 18, 13.5
Part 3	$O_{3,1}$	$O_{3,2}$	Milling	M_2, M_3, M_4	T_7, T_8	20, 20, 15
	$O_{3,2}$	-	Milling	M_2, M_3, M_4	T_7, T_8	20, 20, 15
	$O_{3,3}$	$O_{3,2}$	Milling	M_2, M_3, M_4	T_7, T_8	15, 15, 11.25
Part 4	$O_{4,1}$	$O_{4,3}$	Milling	M_2, M_3	T_6, T_9	12, 15
	$O_{4,2}$	$O_{4,4}$	Milling	M_2, M_3	T_9, T_{10}	21, 18
	$O_{4,3}$	-	Milling	M_2, M_3	T_3	18, 25
	$O_{4,4}$	-	Milling	M_2, M_3	T_1, T_3	27, 25

The PPS problem herein is to determine a process plan and schedule (Gantt chart is shown in the lower part of Figure 4-1), which provides the information for decision-makers on how, when, and in which sequence to allocate these operations of parts to suitable manufacturing resources effectively. When determining the process plan, the best practice operation sequence should be decided first. Then, manufacturing resources such as a machine and one tool should be assigned to every operation. All the manufacturing resources are assumed available in this phase. The determination of schedule is to decide the most appropriate moment to execute each operation with competitive resources like machines, tools, and other possible critical resources. Precedence constraints and resource constraints should be satisfied while determining the process plan and schedule. Moreover, this process plan and schedule should also satisfy the optimization objectives (in this case, minimizing the makespan) concurrently while maintaining the feasibility.

The problem can be defined as follows:

(i) Part scheduling: determining how and when to allocate the manufacturing resources to the parts and satisfying the best practice operation sequencing for all the parts.

(ii) Machine and tool selecting: determining the resource selection according to the feature geometry and available machining resources.

The PPS problem subjects to the following assumptions:

A1. Each resource set (a set of resources needed for processing an operation) can only handle one operation at each time;

A2. Each operation is completed before another operation is loaded;

A3. The sequence of the operations of each part complies with manufacturing constraints;

A4. All parts, machines, tools and other possible resources are available at time zero simultaneously;

A5. Each operation is performed on a single resource set, and each resource can only be occupied by one operation at a time;

A6. The time for setup change is considered as part of the operation. The time for a machine change or a tool change follows the same assumption;

A7. Machines are continuously available for production.

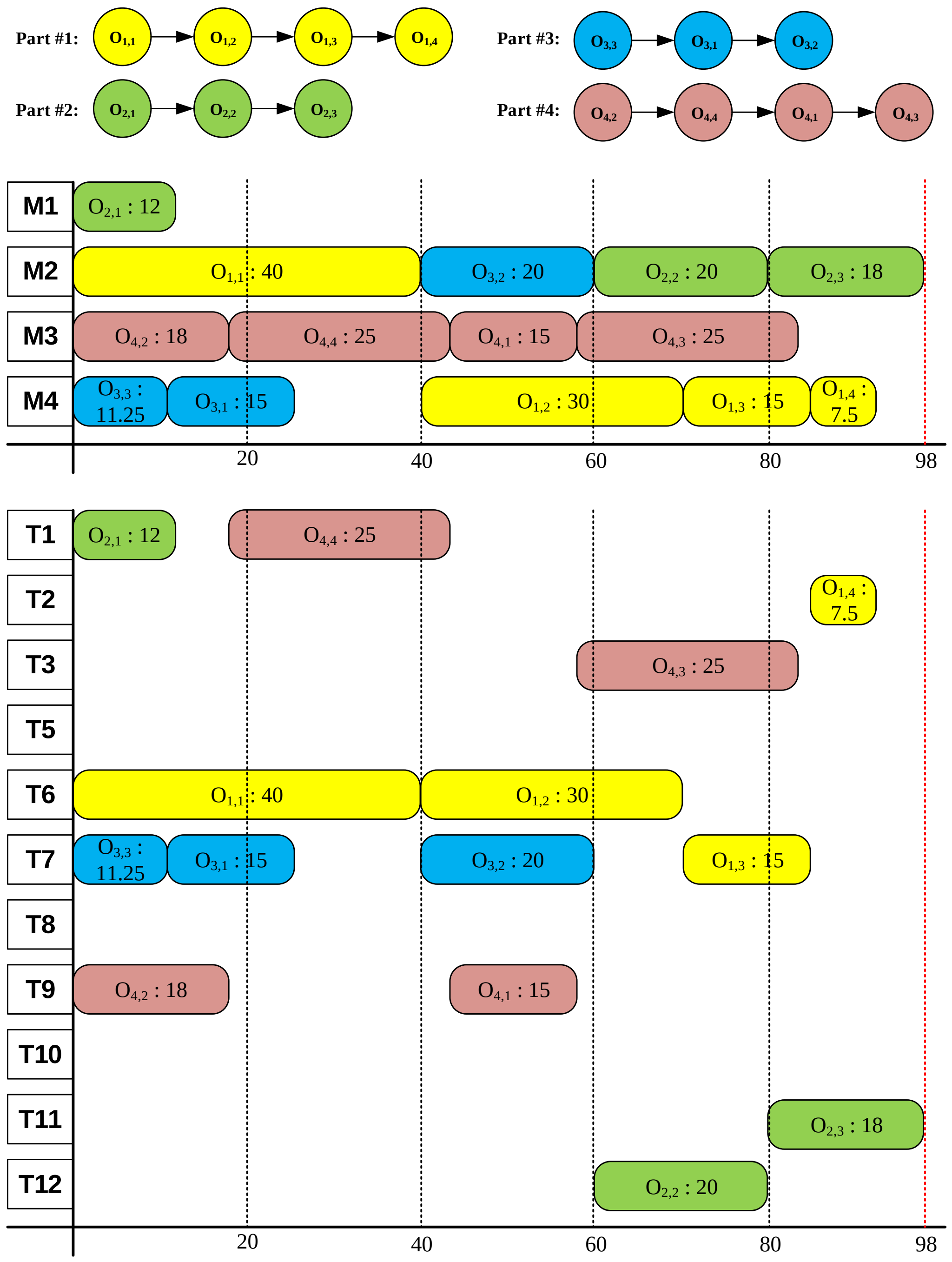


Figure 4-1. Illustration of the PPS Example Problem

As for the constraints, there are precedence constraints among the operations of each part. These precedence relationships must not be violated in the manufacturing process. For example, a best practice operation sequence of 14 operations from example PPS problem is shown as in the top part of Figure 4-1. According to this operation sequence, the manufacturing resources can be specified (machines, tools, and other possible critical resources), and then, the schedule can be determined.

4.2.2 Mathematical Formulation of the PPS Problem

Many important and frequently-used objectives in both literature and real-life are applied in the PPS problem. To name a few, there are minimizing the makespan, variation of workload for each machine, minimizing cost, maximizing capacity utilization, delivery dates, or profit optimizations. In this work, we are focusing on minimizing the makespan as the main objective for our solution to the PPS problem. Minimizing the makespan means that the manufacturing system can get high production in a limited period. Or, in other words, the earliest time for finishing all the planned parts. The mathematical model of the problem is expressed in the following notations:

Indices

i, k: indices of part, $(i, k = 1, 2, \ldots, I)$.

j, h: indices of operation for part i, $(j, h = 1, 2, \ldots, J_i)$.

m: index of machine, $(m = 1, 2, \ldots, M)$.

l: index of tool, $(l = 1, 2, \ldots, L)$.

Parameters

I: number of parts.

J_i: number of operations for part i.

M: number of machines.

L: number of tools.

O_i: set of operations for part i, $O_i = \{o_{i,j} \mid j = 1, 2, \dots, J_i\}$.

$o_{i,j}$: the jth operation of part i.

m_m: the mth machine.

t_l: the lth tool.

$M_{i,j}$: a set of machines that can process $o_{i,j}$.

$L_{i,j}$: a set of tools that can process $o_{i,j}$.

A_m: a set of operations that can be processed on machine m.

A_l: a set of operations that can be processed with tool l.

$r_{i,j,h}$: precedence constraints. if $o_{i,j}$ is predecessor of $o_{i,h}$, $r_{i,j,h} = 1$; otherwise, 0.

$t^P_{m,i,j}$: processing time of $o_{i,j}$ by machine m. All the process related time such as setup time, tool and machine change time are integrated with $t^P_{m,i,j}$.

$t^C_{m,i,j}$: completion time of $o_{i,j}$ by machine m, it should satisfy the inequality $t^C_{m\prime,i,(j-1)} + t^P_{m,i,j} \leq t^C_{m,i,j}$ that means for every operation, its direct predecessor's completion time plus its processing time might be shorter than its completion time.

Decision variables

$$x^M_{m,i,j} = \begin{cases} 1, & if\ o_{i,j}\ is\ performed\ by\ machine\ m, \\ 0, & otherwise \end{cases} \tag{1}$$

$$x_{l,i,j}^{L} = \begin{cases} 1, \ if\ o_{i,j}\ is\ performed\ by\ tool\ l, \\ 0, \ otherwise \end{cases} \tag{2}$$

$$y_{i,j,k,h} = \begin{cases} 1, \ if\ o_{i,j}\ is\ performed\ on\ the\ same\ machine\ right\ before\ o_{k,h}, \\ 0, \ otherwise \end{cases} \tag{3}$$

$$\Omega(X,Y) = \begin{cases} 1, \ if\ X \neq Y, \\ 0, \ otherwise \end{cases} \tag{4}$$

The mathematical model for minimization of makespan can be formulated as the following the mixed-integer programming model:

$$\min t_M = \max_{m,i,j}\{t_{m,i,j}^{C}\} \tag{5}$$

$$s.t. \left(t_{m,k,h}^{C} - t_{m,k,h}^{P} - t_{m,i,j}^{C}\right) * x_{m,i,j}^{M} * x_{m,k,h}^{M} * y_{i,j,k,h} = 0, \forall (i,j),(k,h),m \tag{6}$$

$$s.t. \left(t_{m,k,h}^{C} - t_{m,k,h}^{P} - t_{m,i,j}^{C}\right) * x_{l,i,j}^{L} * x_{l,k,h}^{L} * y_{i,j,k,h} = 0, \forall (i,j),(k,h),l \tag{7}$$

$$r_{i,j,h} * y_{i,h,i,j} = 0, \forall (i,j), h \tag{8}$$

$$y_{i,j,i,j} = 0, \forall (i,j) \tag{9}$$

$$\textstyle\sum_{m-1}^{M} x_{m,i,j}^{M} = 1\,, \forall (i,j) \tag{10}$$

$$\textstyle\sum_{l-1}^{L} x_{l,i,j}^{L} = 1\,, \forall (i,j) \tag{11}$$

$$x_{m,i,j}^{M} = 0, \forall (i,j) \notin A_m, \forall m \tag{12}$$

$$x_{l,i,j}^{L} = 0, \forall (i,j) \notin A_l, \forall l \tag{13}$$

$$y_{i,j,k,h} \in \{0,1\}, \forall (i,j),(k,h) \tag{14}$$

$$x_{m,i,j}^{M} \in \{0,1\}, \forall m,(i,j) \tag{15}$$

$$x_{l,i,j}^{L} \in \{0,1\}, \forall l,(i,j) \tag{16}$$

$$t_{m,i,j}^{C} \geq 0, \forall m,(i,j) \tag{17}$$

$$t_{m,i,j}^{P} \geq 0, \forall m,(i,j) \tag{18}$$

Firstly, the objective function for the PPS problem. Equation (5) illustrates the objective function, which is the minimization of makespan t_M. Makespan t_M is the last operation's finishing time, i.e., the maximization of completion time among all the operations. Secondly, the sequencing constraints. Equations (6) and (7) imposes that any machine or tool cannot be selected for one operation until the predecessor is completed. The precedence constraint is defined as Equation (8). Equation (9) ensures the feasible operation sequence. Thirdly, the incompatible resource constraints. The feasible resource selection is defined by Equations (10) and (11). Equation (10) ensures that one operation is only performed on a single machine, and Equation (11) ensures that one operation requires only one tool. Equation (12) and (13) denotes that the assignment of machine and tool for each operation should be selected from the available machine candidates and tool candidates. Lastly, Equations (14), (15), (16), (17) and (18) impose nonnegative condition.

4.2.3 Discussions on Formulating and Solving the PPS Problem via Conflicting Graph

Based on previous discussions, there are mainly two types of constraints, the sequencing constraints and the incompatible recourse constraints. The former ensures the best practice operation sequence for each part, and the latter ensures no resource conflict for operations scheduled in parallel. Since the operation sequence of the parts is usually predefined, the PPS problem can be considered as selecting the best set of feasible operations that can be processed in parallel during every discrete time period. The feasible operations refer to the operations that can be scheduled for the current time period without resource and precedence conflicts. Usually, there is more than one set of feasible operations can be selected for the current time period. The best set of feasible operations refers to that by scheduling the best set of feasible operations for the current time period, the global optimization objective, minimizing the makespan, is most likely to achieve. If we consider each operation-resource pair (the operation along with one combination of the required resources during a unit discrete time period) as a node, and apply the edges to represent the constraints, a conflicting graph can be generated for the PPS problem. Furthermore, with a weight factor assigned to each operation-resource node as the guidance for selecting the best set of feasible operations, solving the PPS problem becomes solving the MWIS problem for each unit discrete time period. The output of the PPS problem is a combination of the best sets of feasible operations of each unit discrete time period. In the following sections, we discuss how the conflicting graph is generated, how the weight factor is calculated and assigned, and how we generate the optimal or near-optimal solution for the example problem, as shown in Figure 4-1.

4.3 Generating the Conflicting Graph

Based on previous discussions, the PPS problem can be naturally represented as a conflicting graph. Then, the optimization is to find and schedule the best qualified **M**aximal **I**ndependent **S**et (MIS) for each time period, so that an optimal processing schedule can be constructed. In this section, we discuss how to construct the conflicting graph. There are two steps to construct the conflicting graph, Step 1, Operation Data Preparation, and Step 2, Generating the Conflicting Graph.

Step 1. Operation Data Preparation

Before we start to generate the conflicting graph, let us reformulate all operations of the parts that need to be produced. In this step, we need three types of information on the operations of the parts, they are (1) the best practice operation sequence, (2) the resource options of each operation, and (3) the processing time of each operation with each of its resource combinations. The top part of Figure 4-1 illustrates the best practice operation sequence of each part. And from Table 4-1, we understand machine candidates, tool candidates, and machining time associated with the machines, respectively. With this information, we can reformulate the operation information of the parts as Figure 4-2.

Part #1: $O_{1,1} \rightarrow O_{1,2} \rightarrow O_{1,3} \rightarrow O_{1,4}$

$$\begin{pmatrix}\frac{T_{1,1a}[(M_2,\ M_3)_1 \text{ and } (T_6, T_7)_1]}{40}\\ \frac{T_{1,1b}[(M_4)_1 \text{ and } (T_6, T_7)_1]}{30}\end{pmatrix}^1 \rightarrow \begin{pmatrix}\frac{T_{1,2a}[(M_2,\ M_3)_1 \text{ and } (T_6, T_7)_1]}{40}\\ \frac{T_{1,2b}[(M_4)_1 \text{ and } (T_6, T_7)_1]}{30}\end{pmatrix}^1 \rightarrow \begin{pmatrix}\frac{T_{1,3a}[(M_2,\ M_3)_1 \text{ and } (T_6, T_7)_1]}{20}\\ \frac{T_{1,3b}[(M_4)_1 \text{ and } (T_6, T_7)_1]}{15}\end{pmatrix}^1$$

$$\rightarrow \begin{pmatrix}\frac{T_{1,4a}[(M_1)_1 \text{ and } (T_2)_1]}{12}\\ \frac{T_{1,4b}[(M_2, M_3)_1 \text{ and } (T_2)_1]}{10}\\ \frac{T_{1,4c}[(M_4)_1 \text{ and } (T_2)_1]}{7.5}\end{pmatrix}^1$$

Part #2: $O_{2,1} \rightarrow O_{2,2} \rightarrow O_{2,3}$

$$\begin{pmatrix}\frac{T_{2,1a}[(M_1)_1 \text{ and } (T_1)_1]}{12}\\ \frac{T_{2,1b}[(M_2, M_3)_1 \text{ and } (T_1)_1]}{10}\\ \frac{T_{2,1c}[(M_4)_1 \text{ and } (T_1)_1]}{7.5}\end{pmatrix}^1 \rightarrow \begin{pmatrix}\frac{T_{2,2a}[(M_2, M_3)_1 \text{ and } (T_{12})_1]}{20}\\ \frac{T_{2,2b}[(M_4)_1 \text{ and } (T_{12})_1]}{15}\end{pmatrix}^1 \rightarrow \begin{pmatrix}\frac{T_{2,3a}[(M_2, M_3)_1 \text{ and } (T_5, T_6, T_{11})_1]}{18}\\ \frac{T_{2,3b}[(M_4)_1 \text{ and } (T_5, T_6, T_{11})_1]}{13.5}\end{pmatrix}^1$$

Part #3: $O_{3,3} \rightarrow O_{3,1} \rightarrow O_{3,2}$

$$\begin{pmatrix}\frac{T_{3,1a}[(M_2, M_3)_1 \text{ and } (T_7, T_8)_1]}{15}\\ \frac{T_{3,1b}[(M_4)_1 \text{ and } (T_7, T_8)_1]}{11.25}\end{pmatrix}^1 \rightarrow \begin{pmatrix}\frac{T_{3,2a}[(M_2, M_3)_1 \text{ and } (T_7, T_8)_1]}{20}\\ \frac{T_{3,2b}[(M_4)_1 \text{ and } (T_7, T_8)_1]}{15}\end{pmatrix}^1 \rightarrow \begin{pmatrix}\frac{T_{3,3a}[(M_2, M_3)_1 \text{ and } (T_7, T_8)_1]}{20}\\ \frac{T_{3,3b}[(M_4)_1 \text{ and } (T_7, T_8)_1]}{15}\end{pmatrix}^1$$

Part #4: $O_{4,2} \rightarrow O_{4,4} \rightarrow O_{4,1} \rightarrow O_{4,3}$

$$\begin{pmatrix}\frac{T_{4,1a}[(M_2)_1 \text{ and } (T_9, T_{10})_1]}{21}\\ \frac{T_{4,1b}[(M_3)_1 \text{ and } (T_9, T_{10})_1]}{18}\end{pmatrix}^1 \rightarrow \begin{pmatrix}\frac{T_{4,2a}[(M_2)_1 \text{ and } (T_1, T_3)_1]}{27}\\ \frac{T_{4,2b}[(M_3)_1 \text{ and } (T_1, T_3)_1]}{25}\end{pmatrix}^1 \rightarrow \begin{pmatrix}\frac{T_{4,3a}[(M_2)_1 \text{ and } (T_6, T_9)_1]}{12}\\ \frac{T_{4,3b}[(M_3)_1 \text{ and } (T_6, T_9)_1]}{15}\end{pmatrix}^1 \rightarrow$$

$$\begin{pmatrix}\frac{T_{4,4a}[(M_2)_1 \text{ and } (T_3)_1]}{18}\\ \frac{T_{4,4b}[(M_3)_1 \text{ and } (T_3)_1]}{25}\end{pmatrix}^1$$

Figure 4-2. Reformatted Parts Information

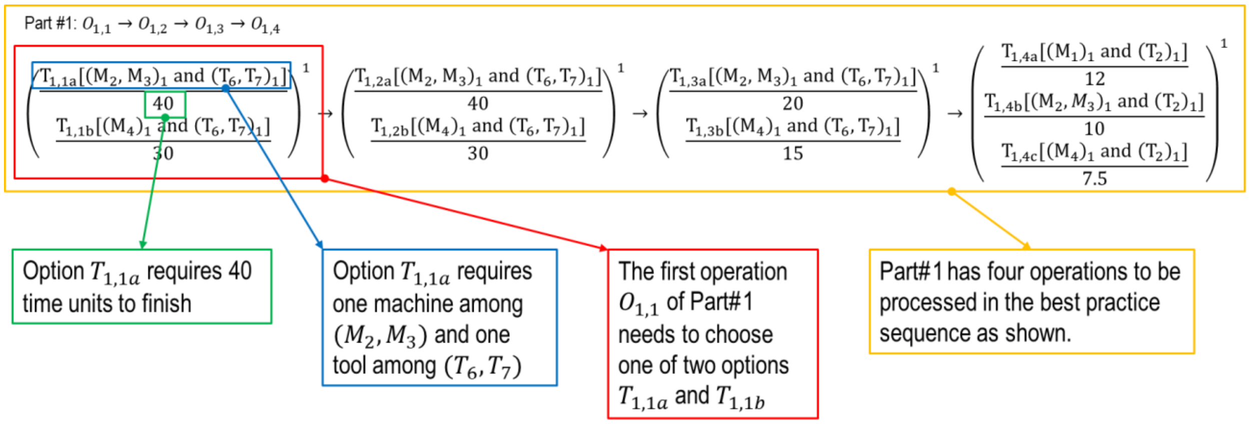

Part #1: $O_{1,1} \rightarrow O_{1,2} \rightarrow O_{1,3} \rightarrow O_{1,4}$
$\left(\frac{T_{1,1a}[(M_2, M_3)_1 \text{ and } (T_6, T_7)_1]}{40} \; \frac{T_{1,1b}[(M_4)_1 \text{ and } (T_6, T_7)_1]}{30}\right)^1 \rightarrow \left(\frac{T_{1,2a}[(M_2, M_3)_1 \text{ and } (T_6, T_7)_1]}{40} \; \frac{T_{1,2b}[(M_4)_1 \text{ and } (T_6, T_7)_1]}{30}\right)^1 \rightarrow \left(\frac{T_{1,3a}[(M_2, M_3)_1 \text{ and } (T_6, T_7)_1]}{20} \; \frac{T_{1,3b}[(M_4)_1 \text{ and } (T_6, T_7)_1]}{15}\right)^1 \rightarrow \left(\frac{T_{1,4a}[(M_1)_1 \text{ and } (T_2)_1]}{12} \; \frac{T_{1,4b}[(M_2, M_3)_1 \text{ and } (T_2)_1]}{10} \; \frac{T_{1,4c}[(M_4)_1 \text{ and } (T_2)_1]}{7.5}\right)^1$
Option $T_{1,1a}$ requires 40 time units to finish
Option $T_{1,1a}$ requires one machine among (M_2, M_3) and one tool among (T_6, T_7)
The first operation $O_{1,1}$ of Part#1 needs to choose one of two options $T_{1,1a}$ and $T_{1,1b}$
Part#1 has four operations to be processed in the best practice sequence as shown.

Figure 4-3. Interpretation for Operation Data Preparation

As described in Figure 4-3, it can be interpreted as the four operations for Part #1 need to be processed in the sequence of $O_{1,1} \rightarrow O_{1,2} \rightarrow O_{1,3} \rightarrow O_{1,4}$. Each operation of each part is corresponding to a detailed task unit. For instance, the first operation $O_{1,1}$is corresponding to the detailed task unit, $\left(\frac{T_{1,1a}[(M_2, M_3)_1 \text{ and } (T_6, T_7)_1]}{40} \; \frac{T_{1,1b}[(M_4)_1 \text{ and } (T_6, T_7)_1]}{30}\right)^1$, which means that operation $O_{1,1}$ can be processed with one of the two task options, $T_{1,1a}$ and $T_{1,1b}$. The $T_{1,1a}$ and $T_{1,1b}$ here indicate that we can choose one of the options "a" or "b" for the operation $O_{1,1}$ as the first operation (task) to produce part #1. The task $T_{1,1a}$ has its detail resource information, $\frac{T_{1,1a}[(M_2, M_3)_1 \text{ and } (T_6, T_7)_1]}{40}$. It means that for the task option $T_{1,1a}$, it requires one of the machines among (M_2, M_3) and one of the tools among (T_6, T_7). And the duration of task option $T_{1,1a}$ is 40 time units.

Each operation with its resource selection needs a certain period of time to process; we can simplify the problem by fitting the processing time of an operation into a discrete number of time slots. For example, if an operation $o_{m,n}$ requires 35 time units to finish, and we define each time slot (1TS) stands for 10 time units. Therefore, the operation $o_{m,n}$ needs 4 time slots (4TS) to process. Based on this assumption, we can translate Figure 4-2 to Figure 4-4 with the simplified processing time (duration) information.

Since we want to use the node in the conflicting graph to represent a task with its resource instance, while choosing the best qualified MIS of nodes, tasks with different durations may cause unbalanced conflicting constraints. Because a long duration task only causes one conflicting count with another conflicting task. In order to capture all the possible constraints, as well as simplify the weight factor calculation and fulfill different weights factor assignment strategies, we want to ensure every node in the conflicting graph stands for one task with one combination instance of its required resources for one time slot. Based on the task information in Figure 4-4, we break down all tasks into single time slots. We name a task that is broken down in such a way as a Unit Task. For example, $\frac{\mathrm{T}_{1,1\mathrm{a}-1}[(\mathrm{M}_2, \mathrm{M}_3)_1 \text{ and } (\mathrm{T}_6,\mathrm{T}_7)_1]}{1TS}$ is a Unit Task, it can be marked as $\mathrm{T}_{1,1\mathrm{a}-1}$, which means that it is the first Unit Task of option "a" in part #1 operations. According to the details of $\mathrm{T}_{1,1\mathrm{a}-1}$, it requires one of the machines among $(\mathrm{M}_2, \mathrm{M}_3)$ and one of the tools among (T_6, T_7). Based on the information from Figure 4-4, the transformed tasks information in Unit Tasks is shown in Figure 4-5. The information in Figure 4-5 can be formulated into a dictionary for the implementation of the proposed approach. The format is shown in Figure 4-6 below; there are 47 Unit Tasks after breaking up. In the next step, we discuss how we can generate the nodes and edges for generating the conflicting graph for the PPS problem.

Part #1: $O_{1,1} \rightarrow O_{1,2} \rightarrow O_{1,3} \rightarrow O_{1,4}$

$$\begin{pmatrix} \dfrac{\mathrm{T}_{1,1a}[(\mathrm{M}_2,\ \mathrm{M}_3)_1 \text{ and } (\mathrm{T}_6, \mathrm{T}_7)_1]}{4TS} \\ \dfrac{\mathrm{T}_{1,1b}[(\mathrm{M}_4)_1 \text{ and } (\mathrm{T}_6, \mathrm{T}_7)_1]}{3TS} \end{pmatrix}^1 \rightarrow \begin{pmatrix} \dfrac{\mathrm{T}_{1,2a}[(\mathrm{M}_2,\ \mathrm{M}_3)_1 \text{ and } (\mathrm{T}_6, \mathrm{T}_7)_1]}{4TS} \\ \dfrac{\mathrm{T}_{1,2b}[(\mathrm{M}_4)_1 \text{ and } (\mathrm{T}_6, \mathrm{T}_7)_1]}{3TS} \end{pmatrix}^1$$

$$\rightarrow \left(\frac{\mathrm{T}_{1,3a}[(\mathrm{M}_2,\ \mathrm{M}_3, \mathrm{M}_4)_1 \text{ and } (\mathrm{T}_6, \mathrm{T}_7)_1]}{2TS}\right)^1 \rightarrow \begin{pmatrix} \dfrac{\mathrm{T}_{1,4a}[(\mathrm{M}_1)_1 \text{ and } (\mathrm{T}_2)_1]}{2TS} \\ \dfrac{\mathrm{T}_{1,4b}[(\mathrm{M}_2,\ \mathrm{M}_3, \mathrm{M}_4)_1 \text{ and } (\mathrm{T}_2)_1]}{1TS} \end{pmatrix}^1$$

Part #2: $O_{2,1} \rightarrow O_{2,2} \rightarrow O_{2,3}$

$$\begin{pmatrix} \dfrac{\mathrm{T}_{2,1b}[(\mathrm{M}_1)_1 \text{ and } (\mathrm{T}_1)_1]}{2TS} \\ \dfrac{\mathrm{T}_{2,1a}[(\mathrm{M}_2,\ \mathrm{M}_3, \mathrm{M}_4)_1 \text{ and } (\mathrm{T}_1)_1]}{1TS} \end{pmatrix}^1 \rightarrow \left(\frac{\mathrm{T}_{2,2a}[(\mathrm{M}_2,\ \mathrm{M}_3, \mathrm{M}_4)_1 \text{ and } (\mathrm{T}_{12})_1]}{2TS}\right)^1$$

$$\rightarrow \left(\frac{\mathrm{T}_{2,3a}[(\mathrm{M}_2,\ \mathrm{M}_3, \mathrm{M}_4)_1 \text{ and } (\mathrm{T}_6, \mathrm{T}_7, \mathrm{T}_{11})_1]}{2TS}\right)^1$$

Part #3: $O_{3,3} \rightarrow O_{3,1} \rightarrow O_{3,2}$

$$\left(\frac{\mathrm{T}_{3,1a}[(\mathrm{M}_2,\ \mathrm{M}_3, \mathrm{M}_4)_1 \text{ and } (\mathrm{T}_7, \mathrm{T}_8)_1]}{2TS}\right)^1 \rightarrow \left(\frac{\mathrm{T}_{3,2a}[(\mathrm{M}_2,\ \mathrm{M}_3, \mathrm{M}_4)_1 \text{ and } (\mathrm{T}_7, \mathrm{T}_8)_1]}{2TS}\right)^1$$

$$\rightarrow \left(\frac{\mathrm{T}_{3,3a}[(\mathrm{M}_2,\ \mathrm{M}_3, \mathrm{M}_4)_1 \text{ and } (\mathrm{T}_7, \mathrm{T}_8)_1]}{2TS}\right)^1$$

Part #4: $O_{4,2} \rightarrow O_{4,4} \rightarrow O_{4,1} \rightarrow O_{4,3}$

$$\begin{pmatrix} \dfrac{\mathrm{T}_{4,1a}[(\mathrm{M}_2)_1 \text{ and } (\mathrm{T}_9, \mathrm{T}_{10})_1]}{3TS} \\ \dfrac{\mathrm{T}_{4,1b}[(\mathrm{M}_3)_1 \text{ and } (\mathrm{T}_9, \mathrm{T}_{10})_1]}{2TS} \end{pmatrix}^1 \rightarrow \left(\frac{\mathrm{T}_{4,2a}[(\mathrm{M}_2, \mathrm{M}_3)_1 \text{ and } (\mathrm{T}_1, \mathrm{T}_3)_1]}{3TS}\right)^1 \rightarrow \left(\frac{\mathrm{T}_{4,3a}[(\mathrm{M}_2, \mathrm{M}_3)_1 \text{ and } (\mathrm{T}_6, \mathrm{T}_9)_1]}{2TS}\right)^1$$

$$\rightarrow \begin{pmatrix} \dfrac{\mathrm{T}_{4,4a}[(\mathrm{M}_2)_1 \text{ and } (\mathrm{T}_3)_1]}{2TS} \\ \dfrac{\mathrm{T}_{4,4b}[(\mathrm{M}_3)_1 \text{ and } (\mathrm{T}_3)_1]}{3TS} \end{pmatrix}^1$$

Figure 4-4. Tasks Information with Simplified Duration Information

Part #1: $O_{1,1} \to O_{1,2} \to O_{1,3} \to O_{1,4}$

$$\left(\begin{matrix} \frac{\mathrm{T}_{1,1a-1}[(\mathrm{M}_2,\ \mathrm{M}_3)_1 \text{ and } (\mathrm{T}_6,\mathrm{T}_7)_1]}{1TS} \to \frac{\mathrm{T}_{1,1a-2}[(\mathrm{M}_2,\ \mathrm{M}_3)_1 \text{ and } (\mathrm{T}_6,\mathrm{T}_7)_1]}{1TS} \to \frac{\mathrm{T}_{1,1a-3}[(\mathrm{M}_2,\ \mathrm{M}_3)_1 \text{ and } (\mathrm{T}_6,\mathrm{T}_7)_1]}{1TS} \to \frac{\mathrm{T}_{1,1a-4}[(\mathrm{M}_2,\ \mathrm{M}_3)_1 \text{ and } (\mathrm{T}_6,\mathrm{T}_7)_1]}{1TS} \\ \frac{\mathrm{T}_{1,1b-1}[(\mathrm{M}_4)_1 \text{ and } (\mathrm{T}_6,\mathrm{T}_7)_1]}{1TS} \to \frac{\mathrm{T}_{1,1b-2}[(\mathrm{M}_4)_1 \text{ and } (\mathrm{T}_6,\mathrm{T}_7)_1]}{1TS} \to \frac{\mathrm{T}_{1,1b-3}[(\mathrm{M}_4)_1 \text{ and } (\mathrm{T}_6,\mathrm{T}_7)_1]}{1TS} \end{matrix}\right)^1$$

$$\to \left(\begin{matrix} \frac{\mathrm{T}_{1,2a-1}[(\mathrm{M}_2,\ \mathrm{M}_3)_1 \text{ and } (\mathrm{T}_6,\mathrm{T}_7)_1]}{1TS} \to \frac{\mathrm{T}_{1,2a-2}[(\mathrm{M}_2,\ \mathrm{M}_3)_1 \text{ and } (\mathrm{T}_6,\mathrm{T}_7)_1]}{1TS} \to \frac{\mathrm{T}_{1,2a-3}[(\mathrm{M}_2,\ \mathrm{M}_3)_1 \text{ and } (\mathrm{T}_6,\mathrm{T}_7)_1]}{1TS} \to \frac{\mathrm{T}_{1,2a-4}[(\mathrm{M}_2,\ \mathrm{M}_3)_1 \text{ and } (\mathrm{T}_6,\mathrm{T}_7)_1]}{1TS} \\ \frac{\mathrm{T}_{1,2b-1}[(\mathrm{M}_4)_1 \text{ and } (\mathrm{T}_6,\mathrm{T}_7)_1]}{1TS} \to \frac{\mathrm{T}_{1,2b-2}[(\mathrm{M}_4)_1 \text{ and } (\mathrm{T}_6,\mathrm{T}_7)_1]}{1TS} \to \frac{\mathrm{T}_{1,2b-3}[(\mathrm{M}_4)_1 \text{ and } (\mathrm{T}_6,\mathrm{T}_7)_1]}{1TS} \end{matrix}\right)^1$$

$$\to \left(\frac{\mathrm{T}_{1,3a-1}[(\mathrm{M}_2,\ \mathrm{M}_3,\mathrm{M}_4)_1 \text{ and } (\mathrm{T}_6,\mathrm{T}_7)_1]}{1TS} \to \frac{\mathrm{T}_{1,3a-2}[(\mathrm{M}_2,\ \mathrm{M}_3,\mathrm{M}_4)_1 \text{ and } (\mathrm{T}_6,\mathrm{T}_7)_1]}{1TS}\right)^1 \to \left(\begin{matrix} \frac{\mathrm{T}_{1,4a-1}[(\mathrm{M}_1)_1 \text{ and } (\mathrm{T}_2)_1]}{1TS} \to \frac{\mathrm{T}_{1,4a-2}[(\mathrm{M}_1)_1 \text{ and } (\mathrm{T}_2)_1]}{1TS} \\ \frac{\mathrm{T}_{1,4b}[(\mathrm{M}_2,\ \mathrm{M}_3,\mathrm{M}_4)_1 \text{ and } (\mathrm{T}_2)_1]}{1TS} \end{matrix}\right)^1$$

Part #2: $O_{2,1} \to O_{2,2} \to O_{2,3}$

$$\left(\begin{matrix} \frac{\mathrm{T}_{2,1b-1}[(\mathrm{M}_1)_1 \text{ and } (\mathrm{T}_1)_1]}{1TS} \to \frac{\mathrm{T}_{2,1b-2}[(\mathrm{M}_1)_1 \text{ and } (\mathrm{T}_1)_1]}{1TS} \\ \frac{\mathrm{T}_{2,1a}[(\mathrm{M}_2,\ \mathrm{M}_3,\mathrm{M}_4)_1 \text{ and } (\mathrm{T}_1)_1]}{1TS} \end{matrix}\right)^1 \to \left(\frac{\mathrm{T}_{2,2a-1}[(\mathrm{M}_2,\ \mathrm{M}_3,\mathrm{M}_4)_1 \text{ and } (\mathrm{T}_{12})_1]}{1TS} \to \frac{\mathrm{T}_{2,2a-2}[(\mathrm{M}_2,\ \mathrm{M}_3,\mathrm{M}_4)_1 \text{ and } (\mathrm{T}_{12})_1]}{1TS}\right)^1$$

$$\to \left(\frac{\mathrm{T}_{2,3a-1}[(\mathrm{M}_2,\ \mathrm{M}_3,\mathrm{M}_4)_1 \text{ and } (\mathrm{T}_5,\mathrm{T}_6,\mathrm{T}_{11})_1]}{1TS} \to \frac{\mathrm{T}_{2,3a-2}[(\mathrm{M}_2,\ \mathrm{M}_3,\mathrm{M}_4)_1 \text{ and } (\mathrm{T}_6,\mathrm{T}_7,\mathrm{T}_{11})_1]}{1TS}\right)^1$$

Part #3: $O_{3,3} \to O_{3,1} \to O_{3,2}$

$$\left(\frac{\mathrm{T}_{3,1a-1}[(\mathrm{M}_2,\ \mathrm{M}_3,\mathrm{M}_4)_1 \text{ and } (\mathrm{T}_7,\mathrm{T}_8)_1]}{1TS} \to \frac{\mathrm{T}_{3,1a-2}[(\mathrm{M}_2,\ \mathrm{M}_3,\mathrm{M}_4)_1 \text{ and } (\mathrm{T}_7,\mathrm{T}_8)_1]}{1TS}\right)^1$$

$$\to \left(\frac{\mathrm{T}_{3,2a-1}[(\mathrm{M}_2,\ \mathrm{M}_3,\mathrm{M}_4)_1 \text{ and } (\mathrm{T}_7,\mathrm{T}_8)_1]}{1TS} \to \frac{\mathrm{T}_{3,2a-2}[(\mathrm{M}_2,\ \mathrm{M}_3,\mathrm{M}_4)_1 \text{ and } (\mathrm{T}_7,\mathrm{T}_8)_1]}{1TS}\right)^1$$

$$\to \left(\frac{\mathrm{T}_{3,3a-1}[(\mathrm{M}_2,\ \mathrm{M}_3,\mathrm{M}_4)_1 \text{ and } (\mathrm{T}_7,\mathrm{T}_8)_1]}{1TS} \to \frac{\mathrm{T}_{3,3a-2}[(\mathrm{M}_2,\ \mathrm{M}_3,\mathrm{M}_4)_1 \text{ and } (\mathrm{T}_7,\mathrm{T}_8)_1]}{1TS}\right)^1$$

Part #4: $O_{4,2} \to O_{4,4} \to O_{4,1} \to O_{4,3}$

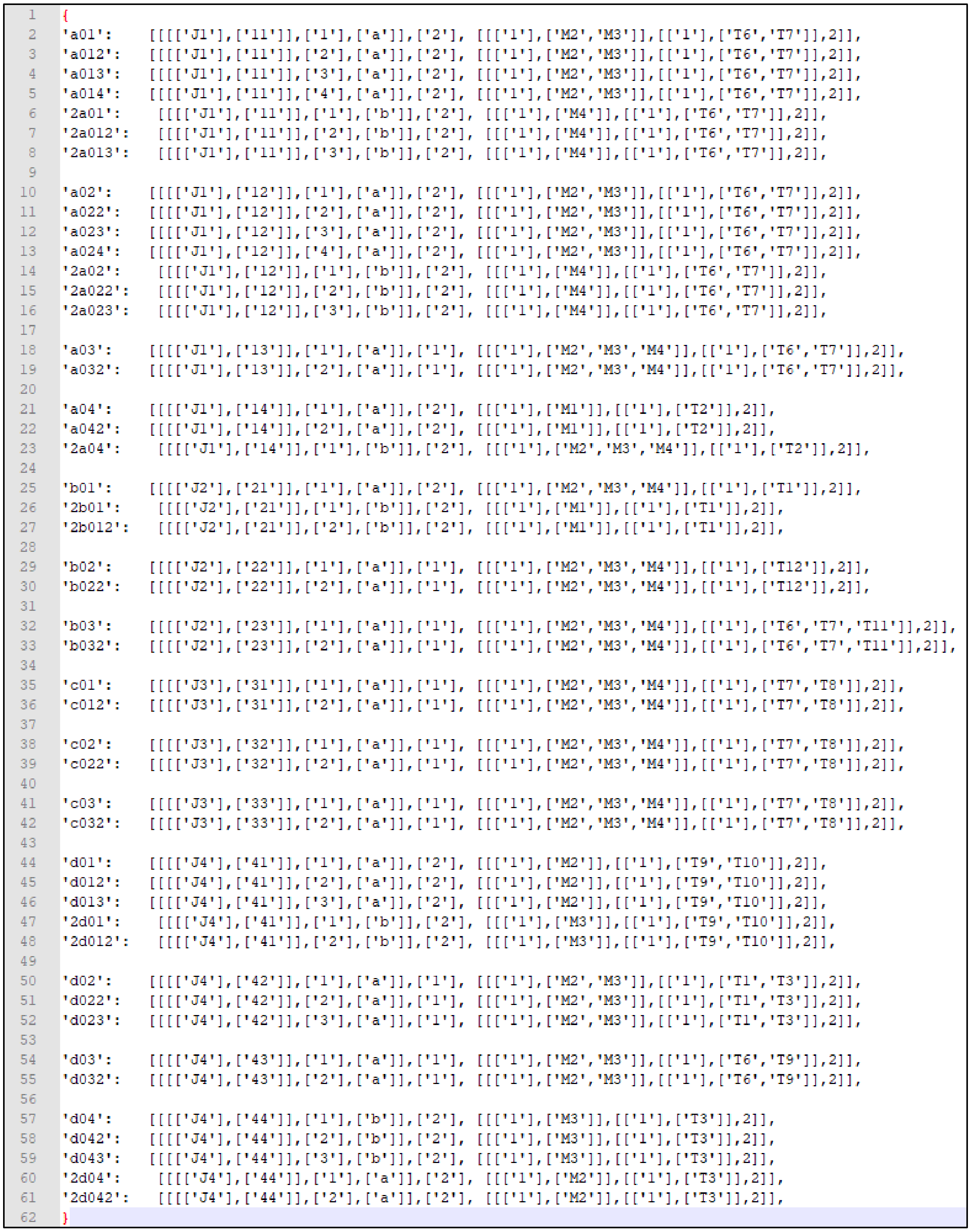
$$\left(\begin{matrix} \frac{\mathrm{T}_{4,1a-1}[(\mathrm{M}_2)_1 \text{ and } (\mathrm{T}_9,\mathrm{T}_{10})_1]}{1TS} \to \frac{\mathrm{T}_{4,1a-2}[(\mathrm{M}_2)_1 \text{ and } (\mathrm{T}_9,\mathrm{T}_{10})_1]}{1TS} \to \frac{\mathrm{T}_{4,1a-3}[(\mathrm{M}_2)_1 \text{ and } (\mathrm{T}_9,\mathrm{T}_{10})_1]}{1TS} \\ \frac{\mathrm{T}_{4,1b-1}[(\mathrm{M}_3)_1 \text{ and } (\mathrm{T}_9,\mathrm{T}_{10})_1]}{1TS} \to \frac{\mathrm{T}_{4,1b-2}[(\mathrm{M}_3)_1 \text{ and } (\mathrm{T}_9,\mathrm{T}_{10})_1]}{1TS} \end{matrix}\right)^1$$

$$\to \left(\frac{\mathrm{T}_{4,2a-1}[(\mathrm{M}_2,\mathrm{M}_3)_1 \text{ and } (\mathrm{T}_1,\mathrm{T}_3)_1]}{1TS} \to \frac{\mathrm{T}_{4,2a-2}[(\mathrm{M}_2,\mathrm{M}_3)_1 \text{ and } (\mathrm{T}_1,\mathrm{T}_3)_1]}{1TS} \to \frac{\mathrm{T}_{4,2a-3}[(\mathrm{M}_2,\mathrm{M}_3)_1 \text{ and } (\mathrm{T}_1,\mathrm{T}_3)_1]}{1TS}\right)^1$$

$$\to \left(\frac{\mathrm{T}_{4,3a-1}[(\mathrm{M}_2,\mathrm{M}_3)_1 \text{ and } (\mathrm{T}_6,\mathrm{T}_9)_1]}{1TS} \to \frac{\mathrm{T}_{4,3a-2}[(\mathrm{M}_2,\mathrm{M}_3)_1 \text{ and } (\mathrm{T}_6,\mathrm{T}_9)_1]}{1TS}\right)^1$$

$$\to \left(\begin{matrix} \frac{\mathrm{T}_{4,4a-1}[(\mathrm{M}_2)_1 \text{ and } (\mathrm{T}_3)_1]}{1TS} \to \frac{\mathrm{T}_{4,4a} - 2[(\mathrm{M}_2)_1 \text{ and } (\mathrm{T}_3)_1]}{1TS} \\ \frac{\mathrm{T}_{4,4b-1}[(\mathrm{M}_3)_1 \text{ and } (\mathrm{T}_3)_1]}{1TS} \to \frac{\mathrm{T}_{4,4b-2}[(\mathrm{M}_3)_1 \text{ and } (\mathrm{T}_3)_1]}{1TS} \to \frac{\mathrm{T}_{4,4b-3}[(\mathrm{M}_3)_1 \text{ and } (\mathrm{T}_3)_1]}{1TS} \end{matrix}\right)^1$$

Figure 4-5. Transformed Tasks Information in Unit Tasks



```
{
'a01':    [[[['J1'],['11']],['1'],['a']],['2'], [[['1'],['M2','M3']],[['1'],['T6','T7']],2]],
'a012':   [[[['J1'],['11']],['2'],['a']],['2'], [[['1'],['M2','M3']],[['1'],['T6','T7']],2]],
'a013':   [[[['J1'],['11']],['3'],['a']],['2'], [[['1'],['M2','M3']],[['1'],['T6','T7']],2]],
'a014':   [[[['J1'],['11']],['4'],['a']],['2'], [[['1'],['M2','M3']],[['1'],['T6','T7']],2]],
'2a01':    [[[['J1'],['11']],['1'],['b']],['2'], [[['1'],['M4']],[['1'],['T6','T7']],2]],
'2a012':   [[[['J1'],['11']],['2'],['b']],['2'], [[['1'],['M4']],[['1'],['T6','T7']],2]],
'2a013':   [[[['J1'],['11']],['3'],['b']],['2'], [[['1'],['M4']],[['1'],['T6','T7']],2]],

'a02':    [[[['J1'],['12']],['1'],['a']],['2'], [[['1'],['M2','M3']],[['1'],['T6','T7']],2]],
'a022':   [[[['J1'],['12']],['2'],['a']],['2'], [[['1'],['M2','M3']],[['1'],['T6','T7']],2]],
'a023':   [[[['J1'],['12']],['3'],['a']],['2'], [[['1'],['M2','M3']],[['1'],['T6','T7']],2]],
'a024':   [[[['J1'],['12']],['4'],['a']],['2'], [[['1'],['M2','M3']],[['1'],['T6','T7']],2]],
'2a02':    [[[['J1'],['12']],['1'],['b']],['2'], [[['1'],['M4']],[['1'],['T6','T7']],2]],
'2a022':   [[[['J1'],['12']],['2'],['b']],['2'], [[['1'],['M4']],[['1'],['T6','T7']],2]],
'2a023':   [[[['J1'],['12']],['3'],['b']],['2'], [[['1'],['M4']],[['1'],['T6','T7']],2]],

'a03':    [[[['J1'],['13']],['1'],['a']],['1'], [[['1'],['M2','M3','M4']],[['1'],['T6','T7']],2]],
'a032':   [[[['J1'],['13']],['2'],['a']],['1'], [[['1'],['M2','M3','M4']],[['1'],['T6','T7']],2]],

'a04':    [[[['J1'],['14']],['1'],['a']],['2'], [[['1'],['M1']],[['1'],['T2']],2]],
'a042':   [[[['J1'],['14']],['2'],['a']],['2'], [[['1'],['M1']],[['1'],['T2']],2]],
'2a04':    [[[['J1'],['14']],['1'],['b']],['2'], [[['1'],['M2','M3','M4']],[['1'],['T2']],2]],

'b01':    [[[['J2'],['21']],['1'],['a']],['2'], [[['1'],['M2','M3','M4']],[['1'],['T1']],2]],
'2b01':    [[[['J2'],['21']],['1'],['b']],['2'], [[['1'],['M1']],[['1'],['T1']],2]],
'2b012':   [[[['J2'],['21']],['2'],['b']],['2'], [[['1'],['M1']],[['1'],['T1']],2]],

'b02':    [[[['J2'],['22']],['1'],['a']],['1'], [[['1'],['M2','M3','M4']],[['1'],['T12']],2]],
'b022':   [[[['J2'],['22']],['2'],['a']],['1'], [[['1'],['M2','M3','M4']],[['1'],['T12']],2]],

'b03':    [[[['J2'],['23']],['1'],['a']],['1'], [[['1'],['M2','M3','M4']],[['1'],['T6','T7','T11']],2]],
'b032':   [[[['J2'],['23']],['2'],['a']],['1'], [[['1'],['M2','M3','M4']],[['1'],['T6','T7','T11']],2]],

'c01':    [[[['J3'],['31']],['1'],['a']],['1'], [[['1'],['M2','M3','M4']],[['1'],['T7','T8']],2]],
'c012':   [[[['J3'],['31']],['2'],['a']],['1'], [[['1'],['M2','M3','M4']],[['1'],['T7','T8']],2]],

'c02':    [[[['J3'],['32']],['1'],['a']],['1'], [[['1'],['M2','M3','M4']],[['1'],['T7','T8']],2]],
'c022':   [[[['J3'],['32']],['2'],['a']],['1'], [[['1'],['M2','M3','M4']],[['1'],['T7','T8']],2]],

'c03':    [[[['J3'],['33']],['1'],['a']],['1'], [[['1'],['M2','M3','M4']],[['1'],['T7','T8']],2]],
'c032':   [[[['J3'],['33']],['2'],['a']],['1'], [[['1'],['M2','M3','M4']],[['1'],['T7','T8']],2]],

'd01':    [[[['J4'],['41']],['1'],['a']],['2'], [[['1'],['M2']],[['1'],['T9','T10']],2]],
'd012':   [[[['J4'],['41']],['2'],['a']],['2'], [[['1'],['M2']],[['1'],['T9','T10']],2]],
'd013':   [[[['J4'],['41']],['3'],['a']],['2'], [[['1'],['M2']],[['1'],['T9','T10']],2]],
'2d01':    [[[['J4'],['41']],['1'],['b']],['2'], [[['1'],['M3']],[['1'],['T9','T10']],2]],
'2d012':   [[[['J4'],['41']],['2'],['b']],['2'], [[['1'],['M3']],[['1'],['T9','T10']],2]],

'd02':    [[[['J4'],['42']],['1'],['a']],['1'], [[['1'],['M2','M3']],[['1'],['T1','T3']],2]],
'd022':   [[[['J4'],['42']],['2'],['a']],['1'], [[['1'],['M2','M3']],[['1'],['T1','T3']],2]],
'd023':   [[[['J4'],['42']],['3'],['a']],['1'], [[['1'],['M2','M3']],[['1'],['T1','T3']],2]],

'd03':    [[[['J4'],['43']],['1'],['a']],['1'], [[['1'],['M2','M3']],[['1'],['T6','T9']],2]],
'd032':   [[[['J4'],['43']],['2'],['a']],['1'], [[['1'],['M2','M3']],[['1'],['T6','T9']],2]],

'd04':    [[[['J4'],['44']],['1'],['b']],['2'], [[['1'],['M3']],[['1'],['T3']],2]],
'd042':   [[[['J4'],['44']],['2'],['b']],['2'], [[['1'],['M3']],[['1'],['T3']],2]],
'd043':   [[[['J4'],['44']],['3'],['b']],['2'], [[['1'],['M3']],[['1'],['T3']],2]],
'2d04':    [[[['J4'],['44']],['1'],['a']],['2'], [[['1'],['M2']],[['1'],['T3']],2]],
'2d042':   [[[['J4'],['44']],['2'],['a']],['2'], [[['1'],['M2']],[['1'],['T3']],2]],
}
```

Figure 4-6. Scheduling Problem Input Format

Step 2. Generating Nodes and Edges of the Conflicting Graph

A conflicting graph consists of two essentials, the nodes and edges. A node is representing one possible resource combination instance of a Unit Task. And the edges are representing the resource constraints of the instances of the Unit Tasks.

Step 2.1 Generating the Nodes

In order to explain how to generate nodes for the conflicting graph, let us take a Unit Task example from Figure 4-5, $\mathrm{T}_{2,1\mathrm{b}-1}$, which is the first Unit Task in option "b" of the first operation in part #2 production processes. Based on the details, $\frac{\mathrm{T}_{2,1\mathrm{b}-1}[(\mathrm{M}_1)_1 \text{ and } (\mathrm{T}_1)_1]}{1TS}$, of this Unit Task, it can be represented by one node, because it only has one possible resource instance, machine M_1 and tool T_1. On the same idea, all the nodes stand for all the possible resource instance of all the Unit Tasks can be generated for the conflicting graph. The node details of the example problem are shown in the first two columns in Figure 4-9.

Step 2.2 Generating the Edges

We developed the following four rules for generating edges in the conflicting graph.

(1) For any two nodes from the same Unit Task, they are connected by an edge. It implies the constraint that for each Unit Task, it can only be scheduled once.
(2) For any two nodes from the same operation, if they belong to different task options, they are connected by an edge. It implies the constraint that for each operation, we can only schedule it with only one task option.
(3) For any two nodes from the same operation and the same task option, but different Unit

Task, if their resources are not the same, they are connected by an edge. It implies the constraint that once an operation is started, the resources have been selected cannot be changed until it is finished.

(4) For the nodes from different parts, if any of their resources is the same, they are connected by an edge. It implies the resource constraints that one resource can be occupied by only one operation during the same time period.

Besides the rules mentioned above, note that there are no edges between the nodes of two different operations for the same part because they cannot be scheduled in the same time slot, and the selection has no effect on each other. This situation is ensured by the weight assignment strategies, which are discussed in detail in the following sections.

To better illustrate the rules for generating the edges of the conflicting graph, let us take the two operations $O_{2,1} \rightarrow O_{2,2}$ $(T_{2,1} \rightarrow T_{2,2})$ of Part #2 from the example problem plus a given operation $O_{i,1}$ $(T_{i,1})$ of Part #i, tasks details are shown as below:

$$\left(\begin{matrix} \dfrac{\mathrm{T}_{2,1a}[(\mathrm{M}_2,\ \mathrm{M}_3, \mathrm{M}_4)_1 \text{ and } (\mathrm{T}_1)_1]}{1TS} \\ \dfrac{\mathrm{T}_{2,1b-1}[(\mathrm{M}_1)_1 \text{ and } (\mathrm{T}_1)_1]}{1TS} \rightarrow \dfrac{\mathrm{T}_{2,1b-2}[(\mathrm{M}_1)_1 \text{ and } (\mathrm{T}_1)_1]}{1TS} \end{matrix}\right)^1$$

$$\rightarrow \left(\dfrac{\mathrm{T}_{2,2a-1}[(\mathrm{M}_2,\ \mathrm{M}_3, \mathrm{M}_4)_1 \text{ and } (\mathrm{T}_{12})_1]}{1TS} \rightarrow \dfrac{\mathrm{T}_{2,2a-2}[(\mathrm{M}_2,\ \mathrm{M}_3, \mathrm{M}_4)_1 \text{ and } (\mathrm{T}_{12})_1]}{1TS}\right)^1$$

And

$$\left(\dfrac{\mathrm{T}_{i,1a-1}[(\mathrm{M}_4)_1 \text{ and } (\mathrm{T}_6)_1]}{1TS}\right)^1$$

A conflict graph can be constructed, as shown in Figure 4-7. The colors differentiate the Unit Tasks, and the numbers on edges indicate the rule used while generating the edges.

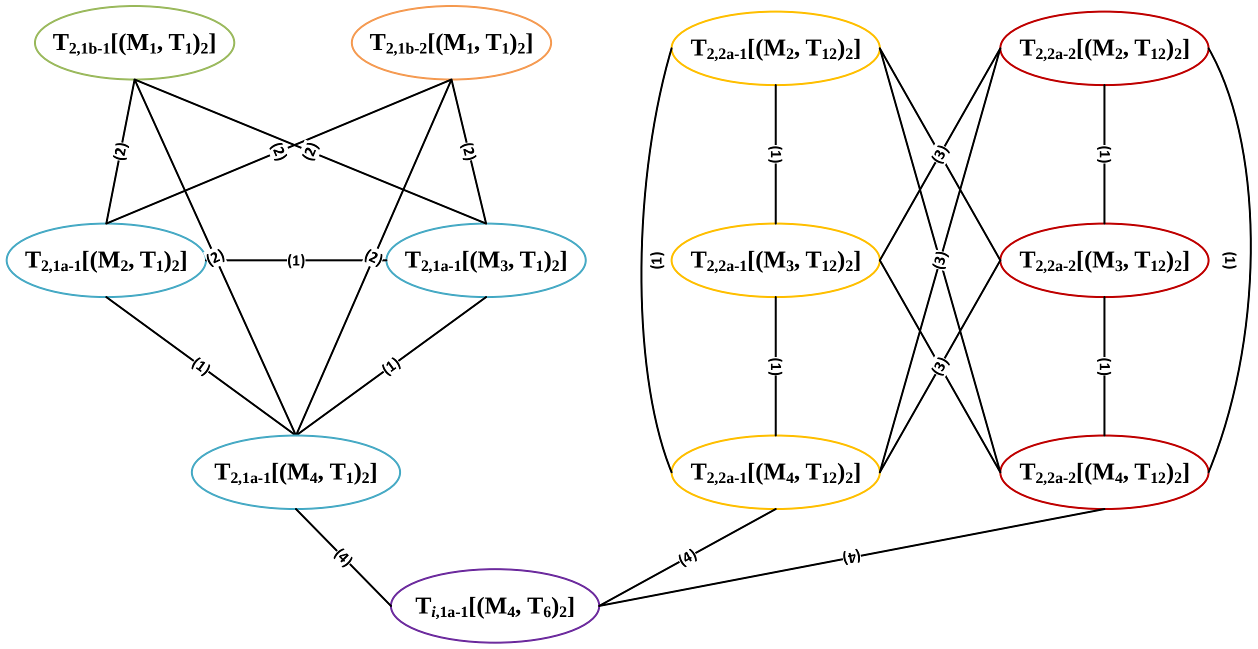


Figure 4-7. The Conflict Graph of operations $O_{2,1} \rightarrow O_{2,2}$ of Part #2 and the operation $O_{i,1}$ $(T_{i,1})$ of Part #i

On the same idea, a conflicting graph for all four parts in the example problem is constructed as Figure 4-8. The graph has 161 nodes and 4718 edges. The node labels and the connection details of the conflicting graph are shown in Figure 4-9. For example, the node '0' represents $\mathrm{T}_{1,1a-1}(M_2, T_6)$, which is one of the resource selections of the Unit Task, $\frac{\mathrm{T}_{1,1a-1}[(\mathrm{M}_2,\mathrm{M}_3)_1 \text{ and } (\mathrm{T}_6,\mathrm{T}_7)_1]}{1TS}$. Note that the color clusters in Figure 4-9 are for differentiating different operations. With the conflicting graph ready, in the next section, we explain how we generate weights for Unit Tasks and how we assign weight factors to nodes so that the MWIS algorithms can be configured to schedule the nodes to achieve the objective of minimizing the makespan for the PPS problem.

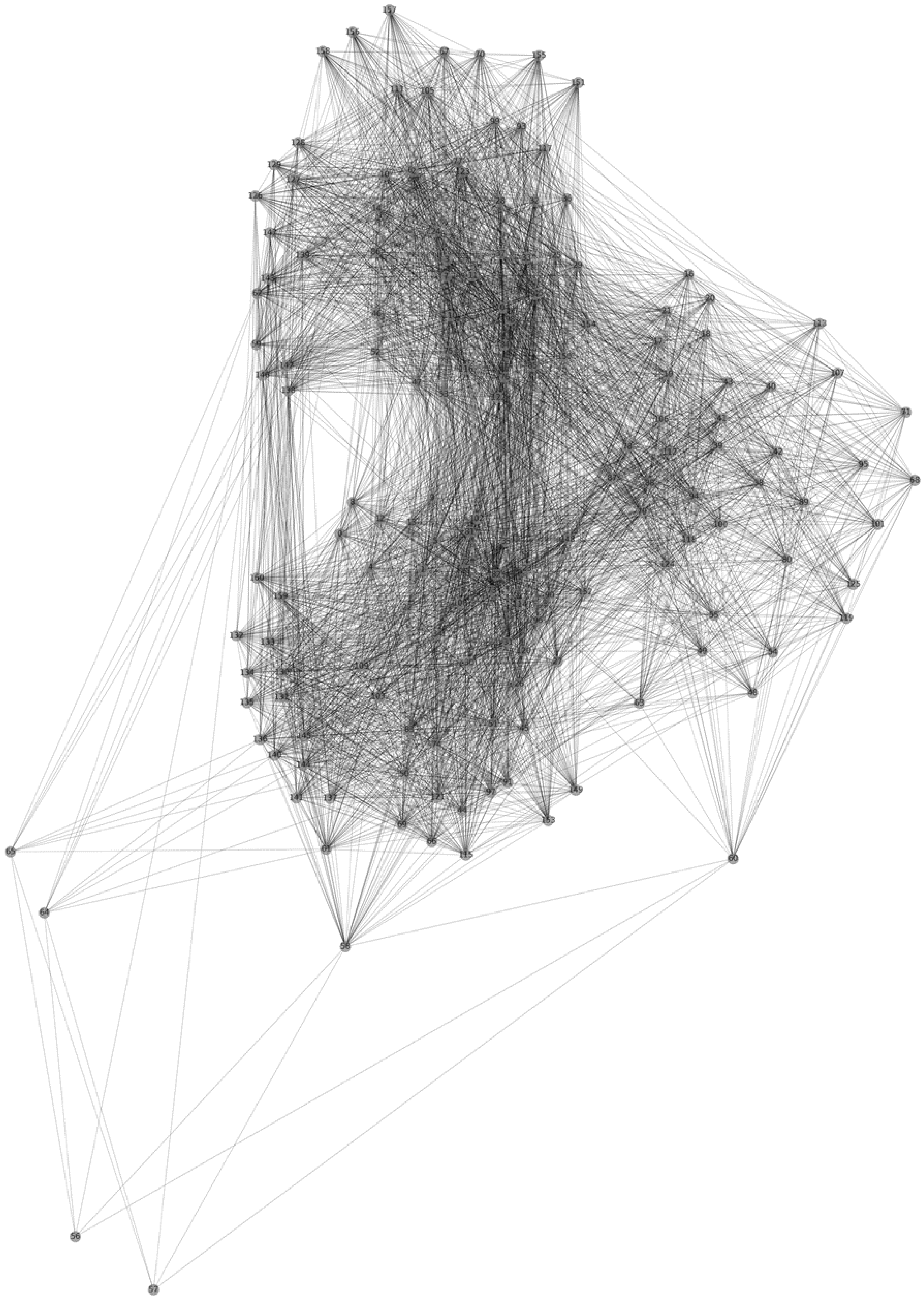


Figure 4-8. Conflict Graph for the Example Problem

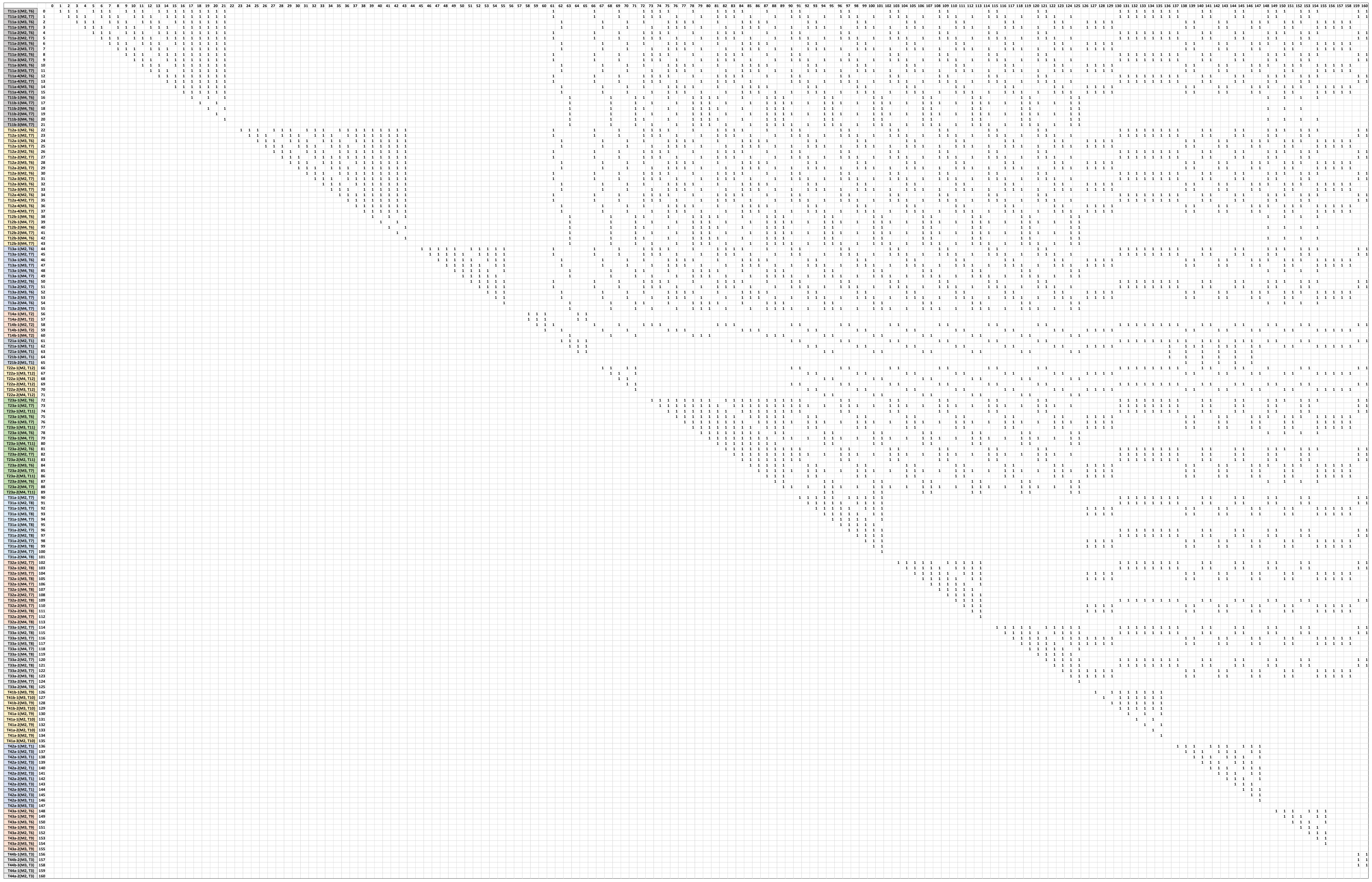


Figure 4-9. Graph Connection Details for the Example Problem

4.4 Weight Factors Calculation and the Configurations of MWIS Algorithms

With the problem formulated as a conflicting graph, our goal is to find the nodes to schedule for each time slot towards the objective of minimizing the total number of required time slots to finish all the operations. The weight factors assigned to nodes of the conflicting graph are used as the guidance for task and resource selections towards the optimal solution of the PPS problem. From Figure 4-4, only the node from the first Unit Task of each option of each part can be scheduled for the current time slot. We name such a Unit Task as a Unit Task Candidate, and the nodes from Unit Task Candidates as Candidate Nodes. The simple idea is that we want to schedule as much as possible Unit Task Candidates at each time slot, and we want to ensure that these scheduled Unit Task Candidates have the most constraints for the rest of Unit Tasks. Because once a Unit Task is scheduled for the current time slot, it is removed from the graph of the following procedures. By doing so, we can remove as many as possible Candidate Nodes at each time slot, and if we can ensure that by removing those nodes, we can remove the most constraints for the remaining Unit Tasks. By discharging the constraints at each time slot, we have more freedom to schedule more Unit Task Candidates in the following time slots. In this sense, we can achieve the optimal or near-optimal result of the PPS problem. In order to execute this idea, we developed a set of heuristics to generate the weights and configure these heuristics with MWIS algorithms discussed in Chapter 3. We are focusing on the weights calculation and the MWIS algorithm configurations for the PPS problem in the following discussions of this section. In Chapter 5, computational experiments are performed on both a real-world case and randomized cases to exam the proposed approach.

4.4.1 The Weights Calculation

From the edges generating rules, Candidate Nodes, which are not compatible due to constraints, are connected. In other words, they are not independent. By applying the MWIS algorithms, we can find the most weighted set of independent candidate nodes, which can be scheduled for the current time slot. We assume that the nodes belong to the same Unit Task should have the same weights. Then, the weight of a Unit Task can be determined based on the conflicting condition of this Unit Task among all Unit Tasks remaining. We can calculate the weights for all Unit Tasks remaining and configure the weight factors for Unit Tasks Candidates with different MWIS algorithms.

We define two types of weight to describe the conflicting condition of a Unit Task.

1. The Unit Task connection weight,

$$W_{connection}(T_{p,t}, T_{p',t'})$$

2. The Unit Task length weight,

$$W_{length}(T_{p,t})$$

   Where the two Unit Tasks, $T_{p,t}$ and $T_{p',t'}$, belong to two different parts, P_p and $P_{p'}$. $p \neq p'$, $t \in [1, t_{p\_max}]$, $t' \in [1, t'_{p'\_max}]$, $p \,\&\, p' \in [1, p_{max}]$, where t_{p_max} is the last Unit Task (the task with the greatest index) in part P_p and $t'_{p'\_max}$ is the last Unit Task in part $P_{p'}$. p_{max} is the index of the last part (the part with the greatest index).

The Unit Task connection weight, $W_{connection}(T_{p,t}, T_{p',t'})$, is based on the connection rate between two Unit Tasks $T_{p,t}$ and $T_{p',t'}$ from different parts P_p and $P_{p'}$. Being inspired by the graph density definition. Let $N(T_{p,t})$ be a set of $n(T_{p,t})$ number of nodes from the Unit Task $T_{p,t}$, and $e(T_{p,t}, T_{p',t'})$ is the number of edges between set $N(T_{p,t})$ and set $N(T_{p',t'})$. Then, we have the Unit Task connection weights:

$$W_{connection}(T_{p,t}, T_{p',t'}) = \frac{e(T_{p,t}, T_{p',t'})}{n(T_{p,t}) * n(T_{p',t'})}$$

The Unit Task length weight, $W_{length}(T_{p,t})$, is the length weight coefficient, LW_c, multiply by the number, $r(T_{p,t})$, of remaining time slots needed to finish part P_p. Where we have:

$$W_{length}(T_{p,t}) = LW_c * r(T_{p,t})$$

Note that the length weight coefficient, LW_c, is used to describe the level priority given to a Unit Task based on the number of time slots remaining for finishing the part individually. Based on our testing, we define three levels of length weight coefficient, median, high and low, as LW_c^M, LW_c^H and LW_c^L respectively. And they are defined as follows:

- Let $LW_c^M = 1$, to keep the length weight coefficient in the same scale as the Unit Task connection weight. In this case, the resource constraints and sequencing constraints are considered as equal while selecting nodes.
- Let $LW_c^H = p_{max} + \sum_{p=1}^{p=p_{max}} r(T_{p,1})$, which is the total number of time slots of all the parts, to ensure the parts need more remaining time slots are given priority.
- $LW_c^L = 0.01$, to keep the length weight coefficient a minimum effect on node selection. In this case, the resource constraints are more emphasized compare to the sequencing constraints while selecting nodes.

The total weight, $W_{total}(T_{p,t})$, of the nodes of a Unit Task, $T_{p,t}$, is the sum of the Unit Task connection weight between itself and all other Unit Tasks of different parts, plus the Unit Task length weight. Formally,

$$W_{total}(T_{p,t}) = W_{length}(T_{p,t}) + \sum_{p'=1,t'=1}^{p'=p_{max},t'=t_{p\_max}} W_{connection}(T_{p,t}, T_{p',t'})$$

Note that the total weight, $W_{total}(T_{p,t})$, as the initial weight value, its purpose is to describe the conflicts that a Unit Task can possibly cause in a PPS problem. The final weight factors need to be configured with the MWIS algorithms for solving the PPS problem. An instance of the weights of the example problem can be calculated as Table 4-2 below. Each column describes the part ID, operation ID, Unit Tasks, nodes, and the value of the initial weights, respectively. Note that in Table 4-2, we choose to use the high length weight coefficient, $LW_c^H = 32$.

Table 4-2. Unit Tasks and Nodes

Part-ID	**Op-ID**	**Unit Tasks**	**Nodes**	**Initial Weights**
Part 1	$O_{1,1}$	$T_{1,1a-1}$	0, 1, 2, 3	157.097
		$T_{1,1a-2}$	4, 5, 6, 7	141.097
		$T_{1,1a-3}$	8, 9, 10, 11	125.097
		$T_{1,1a-4}$	12, 13, 14, 15	117.097
		$T_{1,1b-1}$	16, 17	154.722
		$T_{1,1b-2}$	18, 19	138.722
		$T_{1,1b-3}$	20, 21	122.722
	$O_{1,2}$	$T_{1,2a-1}$	22, 23, 24, 25	101.097
		$T_{1,2a-2}$	26, 27, 28, 29	85.097
		$T_{1,2a-3}$	30, 31, 32, 33	69.097
		$T_{1,2a-4}$	34, 35, 36, 37	61.097
		$T_{1,2b-1}$	38, 39	98.722
		$T_{1,2b-2}$	40, 41	82.722
		$T_{1,2b-3}$	42, 43	66.722
	$O_{1,3}$	$T_{1,3a-1}$	44, 45, 46, 47, 48, 49	88.611
		$T_{1,3a-2}$	50, 51, 52, 53, 54, 55	56.611
	$O_{1,4}$	$T_{1,4a-1}$	56	8.5
		$T_{1,4a-2}$	57	0.5
		$T_{1,4b-1}$	58, 59, 60	11.417
Part 2	$O_{2,1}$	$T_{2,1a-1}$	61, 62, 63	76.750
		$T_{2,1b-1}$	64	73.25
		$T_{2,1b-2}$	65	65.25
	$O_{2,2}$	$T_{2,2a-1}$	66, 67, 68	104.499
		$T_{2,2a-2}$	69, 70, 71	72.499
	$O_{2,3}$	$T_{2,3a-1}$	72, 73, 74, 75, 76, 77, 78, 79, 80	43.389
		$T_{2,3a-2}$	81, 82, 83, 84, 85, 86, 87, 88, 89	11.389
Part 3	$O_{3,3}$	$T_{3,1a-1}$	90, 91, 92, 93, 94, 95	169.722
		$T_{3,1a-2}$	96, 97, 98, 99, 100, 101	137.722
	$O_{3,1}$	$T_{3,2a-1}$	102, 103, 104, 105, 106, 107	105.722
		$T_{3,2a-2}$	108, 109, 110, 111, 112, 113	73.722
	$O_{3,2}$	$T_{3,3a-1}$	114, 115, 116, 117, 118, 119	41.722
		$T_{3,3a-2}$	120, 121, 122, 123, 124, 125	9.722
Part 4	$O_{4,2}$	$T_{4,1b-1}$	126, 127	147.167
		$T_{4,1b-2}$	128, 129	131.167
		$T_{4,1a-1}$	130, 131	147.167

	$T_{4,1a-2}$	132, 133	131.167
	$T_{4,1a-3}$	134, 135	123.167
$O_{4,4}$	$T_{4,2a-1}$	136, 137, 138, 139	215.0
	$T_{4,2a-2}$	140, 141, 142, 143	183.0
	$T_{4,2a-3}$	144, 145, 146, 147	151.0
$O_{4,1}$	$T_{4,3a-1}$	148, 149, 150, 151	120.139
	$T_{4,3a-2}$	152, 153, 154, 155	88.139
$O_{4,3}$	$T_{4,4b-1}$	156	27.167
	$T_{4,4b-2}$	157	11.167
	$T_{4,4b-3}$	158	3.167
	$T_{4,4a-1}$	159	27.168
	$T_{4,4a-2}$	160	11.167

4.4.2 Weight Factor Arrangements with MWIS Algorithms

We have calculated the weight factors for the Unit Tasks, and now we explain how to finalize the weight factors with the MWIS algorithms. We developed three weight factor arrangements for the MWIS-based algorithms and seven weight factor arrangements for the AMISL-based algorithms. The weight factor arrangements, together with the MWIS algorithms, make twenty-eight different heuristics configurations for solving the PPS problem.

Before we start to talk about the weight factor arrangements, let us first recall the eight MWIS algorithms from Chapter 3. These algorithms are:

- Algorithm A1 MWIS: the proposed exact MWIS algorithm.
- Algorithm A2 AMISL: the proposed exact AMISL-based MWIS algorithm.
- Algorithm A3 GWMIN: the GWMIN approximation algorithm from literature.
- Algorithm A4 MWIS_CS_GWMIN: it is an algorithm composed of Algorithm A1 and Algorithm A3. This algorithm computes Compare Sets based on the whole induced subgraph at each level using Algorithm A3 GWMIN.
- Algorithm A5 MWIS_SubCS_GWMIN: it is an algorithm composed of Algorithm A1 and Algorithm A3. This algorithm computes Compare Sets based on the induced CSSs,

excluding the current removed node, using Algorithm A3 GWMIN.

- Algorithm A6 GWMIN2: the GWMIN2 approximation algorithm from literature.
- Algorithm A7 MWIS_CS_GWMIN2: it is an algorithm composed of Algorithm A1 and Algorithm A6. This algorithm computes Compare Sets based on the whole induced subgraph at each level using Algorithm A6 GWMIN2.
- Algorithm A8 MWIS_SubCS_GWMIN2: it is an algorithm composed of Algorithm A1 and Algorithm A6. This algorithm computes Compare Sets based on the induced CSSs, excluding the current removed node, using Algorithm A6 GWMIN2.

The algorithms list above except Algorithm A2 AMISL are MWIS-based algorithms; they require the weights of all nodes to be positive (≥ 0) to make valid comparations in steps so that the final MWIS can be calculated. In this case, the flexibility of weight arrangements is limited, but this is easy to apply approximation strategies to reduce the complexity to speed up the computation. However, Algorithm A2 AMISL first look for all the **M**aximal **I**ndependent **S**ets (MIS), then get the set with the maximum total weight. In this case, the negative and zero weights are allowed. But Algorithm A2 AMISL may have an unreasonable complexity when there is a large number of large size MISs. Algorithm A2 AMISL is also hard to applied approximation strategies. The details of the three weight factor arrangements for the MWIS based algorithms and the seven weight factor arrangements for the AMISL based MWIS algorithms are discussed below. The idea is that while searching for the nodes for the current time slot, the Unit Tasks that can only be scheduled a good number of time slots later may have limited impact. Based on this idea, the wright factor arrangements are created by only checking different limited numbers of steps ahead and aiming to find the best set of the nodes for the current time slot to achieve the objective of minimizing the makespan.

(1) The weight factor arrangements for MWIS based algorithms

For the MWIS based algorithms, we assign weight factors to the Candidate Nodes of Unit Task Candidates according to the three arrangements described below. Then, a small positive value (for instance, 0.0000001) is assigned to the non-candidate nodes. With the weight factors ready, we can apply one of the seven MWIS-based algorithms to find the set of Candidate Nodes with the maximum total weight with the maximum number of nodes. For this setup, the Candidate Nodes associated with the most uncommon resources for the non-candidate nodes are scheduled for the current time slot. So that there are fewer conflicts for the following time slots if the operations scheduled for the current time slot must be continued for more time slots. The Unit Tasks Candidates with the associated resources represented by the set of nodes are scheduled for the current time slot. The three weight factor arrangements are as follows:

(1.1) MWIS A1: MWIS Weights 1

In each time slot, for each $p \in [1, p_{max}]$, assign weight factors to the Unit Task Candidates, $T^{C}_{p,t_{min}}$, which is the first Unit Task, $T_{p,t_{min}}$, that can be scheduled for part P_p. The value of weight factors of the candidate nodes in $T^{C}_{p,t_{min}}$ is the weight of $T_{p,t_{min}}$, as the equation below,

$$W_{MWIS\_A1\_condidate}\left(T^{C}_{p,t_{min}}\right) = W_{total}\left(T_{p,t_{min}}\right)$$

(1.2) MWIS A2: MWIS Weights 2

In each time slot, for each $p \in [1, p_{max}]$, assign weight factors to the Unit Task Candidates, $T^{C}_{p,t_{min}}$. The value of weight factors of the Candidate Nodes in $T^{C}_{p,t_{min}}$ is the sum of the weights of $T_{p,t_{min}}$ and $T_{p,(\,t_{min}+1)}$, where $T_{p,(\,t_{min}+1)}$ is the following Unit Task of $T_{p,t_{min}}$, as the equation below,

$$W_{MWIS\_A2\_condidate}\left(T^{C}_{p,t_{min}}\right) = W_{total}\left(T_{p,t_{min}}\right) + W_{total}\left(T_{p,t_{min}+1}\right)$$

(1.3) MWIS A3: MWIS Weights 3

In each time slot, for each $p \in [1, p_{max}]$, assign weight factors to the Unit Task Candidates, $T^{C}_{p,t_{min}}$. The value of weight factors of the Candidate Nodes in $T^{C}_{p,t_{min}}$ is the sum of the weights of $T_{p,t_{min}}$, $T_{p,(\,t_{min}+1)}$ and $T_{p,(\,t_{min}+2)}$, where $T_{p,(\,t_{min}+2)}$ is the following Unit Task of $T_{p,(\,t_{min}+1)}$, as the equation below,

$$W_{MWIS\_A3\_condidate}\left(T^{C}_{p,t_{min}}\right) = W_{total}\left(T_{p,t_{min}}\right) + W_{total}\left(T_{p,t_{min}+1}\right) + W_{total}\left(T_{p,t_{min}+2}\right)$$

(2) The weight factor arrangements for Algorithm A2 AMISL

For the AMISL based algorithms, we assign weight factors to the nodes indicated by the seven different weight factor arrangements described below. Then, a small negative value (for instance, -0.0000001) is assigned to the unaddressed nodes. With the weight factors ready, applied Algorithm A2 AMISL to find the set of Candidate Nodes with the maximum total weight with the minimum number of nodes. For this setup, the Candidate Nodes associated with the most common resources for the unaddressed nodes are scheduled for the current time slot, so that the most constraints are removed for the following time slots by scheduling such a set of Candidate Nodes. The Unit Tasks Candidates with the associated resources represented by the set of nodes are scheduled for the current time slot. The seven weight factor arrangements are as follows:

(2.1) AMISL A1: AMISL Weights 1

In each time slot, for each $p \in [1, p_{max}]$, assign weight factors to the Unit Task Candidates, $T^{C}_{p,t_{min}}$. The value of weight factors of the Candidate Nodes in $T^{C}_{p,t_{min}}$ is the weight of $T_{p,t_{min}}$, as the equation below,

$$W_{AMISL\_A1\_condidate}\left(T^{C}_{p,t_{min}}\right) = W_{total}\left(T_{p,t_{min}}\right)$$

(2.2) AMISL A2: AMISL Weights 2 Aggregation

In each time slot, for each $p \in [1, p_{max}]$, assign weight factors to the Unit Task Candidates, $T^{C}_{p,t_{min}}$. The value of weight factors of the Candidate Nodes in $T^{C}_{p,t_{min}}$ is the sum of the weights of $T_{p,t_{min}}$ and $T_{p,(\,t_{min}+1)}$, where $T_{p,(\,t_{min}+1)}$ is the following Unit Task of $T_{p,t_{min}}$, as the equation below,

$$W_{AMISL\_A2\_condidate}\left(T^{C}_{p,t_{min}}\right) = W_{total}\left(T_{p,t_{min}}\right) + W_{total}\left(T_{p,t_{min}+1}\right)$$

(2.3) AMISL A3: AMISL Weights 2 Aggregation + Non-aggregation

In each time slot, for each $p \in [1, p_{max}]$, assign weight factors to the Unit Task Candidates, $T^{C}_{p,t_{min}}$, and the following Unit Task, $T_{p,(\,t_{min}+1)}$. The value of weight factors of the Candidate Nodes in $T^{C}_{p,t_{min}}$ is the sum of the weights of $T_{p,t_{min}}$ and $T_{j,(\,t_{min}+1)}$, as the equation below,

$$W_{AMISL\_A3\_condidate}\left(T^{C}_{p,t_{min}}\right) = W_{total}\left(T_{p,t_{min}}\right) + W_{total}\left(T_{p,t_{min}+1}\right)$$

The value of weight factors of nodes in $T_{p,t_{min}+1}$ is the weight of $T_{p,t_{min}+1}$, as the equation below,

$$W_{AMISL\_A3\_following}\left(T_{p,t_{min}+1}\right) = W_{total}\left(T_{p,t_{min}+1}\right)$$

(2.4) AMISL A4: AMISL Weights 2 Non-aggregation

In each time slot, for each $p \in [1, p_{max}]$, assign weight factors to the Unit Task Candidates, $T^{C}_{p,t_{min}}$, and the following Unit Task, $T_{p,(\,t_{min}+1)}$. The value of weight factors of the Candidate Nodes in $T^{C}_{p,t_{min}}$ is the weight of $T_{p,t_{min}}$, as the equation below,

$$W_{AMISL\_A4\_condidate}\left(T^{C}_{p,t_{min}}\right) = W_{total}\left(T_{p,t_{min}}\right)$$

The value of weight factors of nodes in $T_{p,t_{min}+1}$ is the weight of $T_{p,t_{min}+1}$, as the equation below,

$$W_{AMISL\_A4\_following}\left(T_{p,t_{min}+1}\right) = W_{total}\left(T_{p,t_{min}+1}\right)$$

(2.5) AMISL A5: AMISL Weights 3 Aggregation

In each time slot, for each $p \in [1, p_{max}]$, assign weight factors to the Unit Task Candidates, $T^{C}_{p,t_{min}}$. The value of weight factors of the Candidate Nodes in $T^{C}_{p,t_{min}}$ is the sum of the weights of $T_{p,t_{min}}$, $T_{p,(\,t_{min}+1)}$ and $T_{p,(\,t_{min}+2)}$, as the equation below,

$$W_{AMISL\_A5\_condidate}\left(T^{C}_{p,t_{min}}\right) = W_{total}\left(T_{p,t_{min}}\right) + W_{total}\left(T_{p,t_{min}+1}\right) + W_{total}\left(T_{p,t_{min}+2}\right)$$

(2.6) AMISL A6: AMISL Weights 3 Aggregation + Non-aggregation

In each time slot, for each $p \in [1, p_{max}]$, assign weight factors to the Unit Task Candidates, $T^{C}_{p,t_{min}}$, and the two following Unit Tasks, $T_{p,(\,t_{min}+1)}$ and $T_{p,(\,t_{min}+2)}$. The value of weight factors of the Candidate Nodes in $T^{C}_{p,t_{min}}$ is the sum of the weights of $T_{p,t_{min}}$, $T_{p,(\,t_{min}+1)}$ and $T_{p,(\,t_{min}+2)}$, as the equation below,

$$W_{MWIS\_A6\_condidate}\left(T^{C}_{p,t_{min}}\right) = W_{total}\left(T_{p,t_{min}}\right) + W_{total}\left(T_{p,t_{min}+1}\right) + W_{total}\left(T_{p,t_{min}+2}\right)$$

The value of weight factors of nodes in $T_{p,t_{min}+1}$ is the weight of $T_{p,t_{min}+1}$, as the equation below,

$$W_{AMISL\_A6\_following}\left(T_{p,t_{min}+1}\right) = W_{total}\left(T_{p,t_{min}+1}\right)$$

The value of weight factors of nodes in $T_{p,t_{min}+2}$ is the weight of $T_{p,t_{min}+2}$, as the equation below,

$$W_{AMISL\_A6\_following\_two}\left(T_{p,t_{min}+2}\right) = W_{total}\left(T_{p,t_{min}+2}\right)$$

(2.7) AMISL A7: AMISL Weights 3 Non-aggregation

In each time slot, for each $p \in [1, p_{max}]$, assign weight factors to the Unit Task Candidates, $T_{p,t_{min}}^{C}$, and the two following Unit Tasks, $T_{p,(\ t_{min}+1)}$ and $T_{p,(\ t_{min}+2)}$. The value of weight factors of the Candidate Nodes in $T_{p,t_{min}}^{C}$ is the weights of $T_{p,t_{min}}$, as the equation below,

$$W_{MWIS\_A7\_condidate}\left(T_{p,t_{min}}^{C}\right) = W_{total}\left(T_{p,t_{min}}\right)$$

The value of weight factors of nodes in $T_{p,t_{min}+1}$ is the weight of $T_{p,t_{min}+1}$, as the equation below,

$$W_{AMISL\_A7\_following}\left(T_{p,t_{min}+1}\right) = W_{total}\left(T_{p,t_{min}+1}\right)$$

The value of weight factors of nodes in $T_{p,t_{min}+2}$ is the weight of $T_{p,t_{min}+2}$, as the equation below,

$$W_{AMISL\_A7\_following\_two}\left(T_{p,t_{min}+2}\right) = W_{total}\left(T_{p,t_{min}+2}\right)$$

4.4.3 Heuristics Configurations

The eight MWIS algorithms, together with the ten weight arrangements, can be configured into twenty-eight heuristics configurations for solving the PPS problem. The heuristics configurations are shown in Table 4-3. Each column describes the heuristics configuration ID, algorithm ID, weight arrangement strategies, whether it is an MWIS-based algorithm and whether it is an approximation algorithm, respectively.

Table 4-3. Heuristics Configurations

Heuristics-ID	**Algorithm-ID**	**Weight arrangement strategies**	**MWIS based?**	**Appr?**
1	A1 MWIS	MWIS A1: MWIS Weights 1	Yes	No
2	A1 MWIS	MWIS A2: MWIS Weights 2	Yes	No
3	A1 MWIS	MWIS A3: MWIS Weights 3	Yes	No
4	A2 AMISL	AMISL A1: AMISL Weights 1	No	No
5	A2 AMISL	AMISL A2: AMISL Weights 2 Agg	No	No
6	A2 AMISL	AMISL A3: AMISL Weights 2 Agg + Nagg	No	No
7	A2 AMISL	AMISL A4: AMISL Weights 2 Nagg	No	No
8	A2 AMISL	AMISL A5: AMISL Weights 3 Agg	No	No
9	A2 AMISL	AMISL A6: AMISL Weights 3 Agg + Nagg	No	No
10	A2 AMISL	AMISL A7: AMISL Weights 3 Nagg	No	No
11	A3 GWMIN	MWIS A1: MWIS Weights 1	Yes	Yes
12	A4 MWIS_CS_GWMIN	MWIS A1: MWIS Weights 1	Yes	Yes
13	A5 MWIS_SubCS_GWMIN	MWIS A1: MWIS Weights 1	Yes	Yes
14	A3 GWMIN	MWIS A2: MWIS Weights 2	Yes	Yes
15	A4 MWIS_CS_GWMIN	MWIS A2: MWIS Weights 2	Yes	Yes
16	A5 MWIS_SubCS_GWMIN	MWIS A2: MWIS Weights 2	Yes	Yes
17	A3 GWMIN	MWIS A3: MWIS Weights 3	Yes	Yes
18	A4 MWIS_CS_GWMIN	MWIS A3: MWIS Weights 3	Yes	Yes
19	A5 MWIS_SubCS_GWMIN	MWIS A3: MWIS Weights 3	Yes	Yes
20	A6 GWMIN2	MWIS A1: MWIS Weights 1	Yes	Yes
21	A7 MWIS_CS_GWMIN2	MWIS A1: MWIS Weights 1	Yes	Yes
22	A8 MWIS_SubCS_GWMIN2	MWIS A1: MWIS Weights 1	Yes	Yes
23	A6 GWMIN2	MWIS A2: MWIS Weights 2	Yes	Yes
24	A7 MWIS_CS_GWMIN2	MWIS A2: MWIS Weights 2	Yes	Yes
25	A8 MWIS_SubCS_GWMIN2	MWIS A2: MWIS Weights 2	Yes	Yes
26	A6 GWMIN2	MWIS A3: MWIS Weights 3	Yes	Yes
27	A7 MWIS_CS_GWMIN2	MWIS A3: MWIS Weights 3	Yes	Yes
28	A8 MWIS_SubCS_GWMIN2	MWIS A3: MWIS Weights 3	Yes	Yes

4.5 Solving the Example Problem via the Proposed Approach

In this section, we summarize the proposed method for solving the PPS problem with the example PPS problem described at the beginning of this chapter. The major steps of the proposed approach are described below:

Step #1: Prepare the input information.

In step one, we reformat the operation information with the best practice operation sequence and simplify the problem by breaking down the processing time into time slots. The operation information of the four parts in the example problem is reformatted as Figure 4-2 based on operation sequencing constraints shown as the top part of Figure 4-1. Figure 4-4 can be transformed based on Figure 4-2 by breaking down the operations into Unit Tasks. Here, the processing time for each Unit Task is one time slot, which stands for 10 time units.

Step #2: Generate the conflicting graph for the PPS problem.

In step two, the nodes for the conflict graph is generated based on the different possible resource selections for each Unit Task. And the edges of the conflicting graph are generated based on the constraints. Figure 4-8 is the conflicting graph for the example problem, which has 4718 edges and 161 nodes, and Figure 4-9 shows the details of the edges.

Step #3: Based on the selected heuristics configuration, arrange weight factors and compute the MWIS.

In step three, we select Heuristics #13, which assigns weight factors as MWIS A1: MWIS Weights 1 and uses Algorithm A5 MWIS_SubCS_GWMIN to compute the MWIS for the nodes to schedule for the current time slot. Note that we choose to use the high length weight coefficient, LW_c^H, which $LW_c^H = 32$ for the example problem. The final weight factors at the first time slot for the Unit Task Candidates and Candidate Nodes of the example problem are shown in Table 4-4. Each column describes the part ID, operation ID, Unit Tasks, nodes of the Unit Tasks and the Final weight factors, respectively. The MWIS found by Heuristics #13 is the node set, ['0', '4', '8', '12', '22', '26', '30', '34', '44', '50', '139', '143', '147', '126', '128', '151', '155', '156', '157', '158', '58', '95', '101', '107', '113', '119', '125', '64', '65']. It means that the Unit Task Candidates with their resources, $\mathrm{T}_{1,1a-1}[(\mathrm{M}_2)_1 \text{ and } (\mathrm{T}_6)_1]$, $\mathrm{T}_{2,1b-1}[(\mathrm{M}_1)_1 \text{ and } (\mathrm{T}_1)_1]$, $\mathrm{T}_{3,1a-1}[(\mathrm{M}_4)_1 \text{ and } (\mathrm{T}_8)_1]$ and $\mathrm{T}_{4,1b-1}[(\mathrm{M}_3)_1 \text{ and } (\mathrm{T}_9)_1]$, are scheduled for the current time slot.

Step #4: Update the remaining Unit Tasks and the conflicting graph

In step four, remove the Unit Tasks that have been scheduled and remove the Unit Tasks that cannot be scheduled because of the constraints that no changing resources is allowed before an operation is finished. Then, update the conflicting graph and the weight factors. Figure 4-10 is the updated task information for the remaining Unit Tasks. And the updated remaining conflicting graph, the node labels, and edge connection details for the following time slot are shown as Figure 4-11 and Figure 4-12, respectively.

Table 4-4. Final Weight Factors Unit Task Candidates and Candidate Nodes via Heuristics #13 on the Example Problem

Part-ID	**Op-ID**	**Unit Tasks**	**Nodes**	**Final Weights**
Part 1	$O_{1,1}$	$T_{1,1a-1}$	0, 1, 2, 3	157.097
		$T_{1,1a-2}$	4, 5, 6, 7	141.097
		$T_{1,1a-3}$	8, 9, 10, 11	125.097
		$T_{1,1a-4}$	12, 13, 14, 15	117.097
		$T_{1,1b-1}$	16, 17	154.722
		$T_{1,1b-2}$	18, 19	138.722
		$T_{1,1b-3}$	20, 21	122.722
Part 2	$O_{2,1}$	$T_{2,1a-1}$	61, 62, 63	76.750
		$T_{2,1b-1}$	64	73.25
		$T_{2,1b-2}$	65	65.25
Part 3	$O_{3,3}$	$T_{3,1a-1}$	90, 91, 92, 93, 94, 95	169.722
		$T_{3,1a-2}$	96, 97, 98, 99, 100, 101	137.722
Part 4	$O_{4,2}$	$T_{4,1b-1}$	126, 127	147.167
		$T_{4,1b-2}$	128, 129	131.167
		$T_{4,1a-1}$	130, 131	147.167
		$T_{4,1a-2}$	132, 133	131.167
		$T_{4,1a-3}$	134, 135	123.167

Job #1: $O_{11} \rightarrow O_{12} \rightarrow O_{13} \rightarrow O_{14}$

$$\left(\frac{\mathrm{T}_{11a-2}[(\mathrm{M}_2,\ \mathrm{M}_3)_1 \text{ and } (\mathrm{T}_6, \mathrm{T}_7)_1]}{1TS} \rightarrow \frac{\mathrm{T}_{11a-3}[(\mathrm{M}_2,\ \mathrm{M}_3)_1 \text{ and } (\mathrm{T}_6, \mathrm{T}_7)_1]}{1TS} \rightarrow \frac{\mathrm{T}_{11a-4}[(\mathrm{M}_2,\ \mathrm{M}_3)_1 \text{ and } (\mathrm{T}_6, \mathrm{T}_7)_1]}{1TS}\right)^1$$

$$\rightarrow \left(\begin{matrix} \frac{\mathrm{T}_{12a-1}[(\mathrm{M}_2,\ \mathrm{M}_3)_1 \text{ and } (\mathrm{T}_6, \mathrm{T}_7)_1]}{1TS} \rightarrow \frac{\mathrm{T}_{12a-2}[(\mathrm{M}_2,\ \mathrm{M}_3)_1 \text{ and } (\mathrm{T}_6, \mathrm{T}_7)_1]}{1TS} \rightarrow \frac{\mathrm{T}_{12a-3}[(\mathrm{M}_2,\ \mathrm{M}_3)_1 \text{ and } (\mathrm{T}_6, \mathrm{T}_7)_1]}{1TS} \rightarrow \frac{\mathrm{T}_{12a-4}[(\mathrm{M}_2,\ \mathrm{M}_3)_1 \text{ and } (\mathrm{T}_6, \mathrm{T}_7)_1]}{1TS} \\ \frac{\mathrm{T}_{12b-1}[(\mathrm{M}_4)_1 \text{ and } (\mathrm{T}_6, \mathrm{T}_7)_1]}{1TS} \rightarrow \frac{\mathrm{T}_{12b-2}[(\mathrm{M}_4)_1 \text{ and } (\mathrm{T}_6, \mathrm{T}_7)_1]}{1TS} \rightarrow \frac{\mathrm{T}_{12b-3}[(\mathrm{M}_4)_1 \text{ and } (\mathrm{T}_6, \mathrm{T}_7)_1]}{1TS} \end{matrix}\right)^1$$

$$\rightarrow \left(\frac{\mathrm{T}_{13a-1}[(\mathrm{M}_2,\ \mathrm{M}_3, \mathrm{M}_4)_1 \text{ and } (\mathrm{T}_6, \mathrm{T}_7)_1]}{1TS} \rightarrow \frac{\mathrm{T}_{13a-2}[(\mathrm{M}_2,\ \mathrm{M}_3, \mathrm{M}_4)_1 \text{ and } (\mathrm{T}_6, \mathrm{T}_7)_1]}{1TS}\right)^1 \rightarrow \left(\begin{matrix} \frac{\mathrm{T}_{14a-1}[(\mathrm{M}_1)_1 \text{ and } (\mathrm{T}_2)_1]}{1TS} \rightarrow \frac{\mathrm{T}_{14a-2}[(\mathrm{M}_1)_1 \text{ and } (\mathrm{T}_2)_1]}{1TS} \\ \frac{\mathrm{T}_{14b}[(\mathrm{M}_2,\ \mathrm{M}_3, \mathrm{M}_4)_1 \text{ and } (\mathrm{T}_2)_1]}{1TS} \end{matrix}\right)^1$$

Job #2: $O_{21} \rightarrow O_{22} \rightarrow O_{23}$

$$\left(\frac{\mathrm{T}_{21b-2}[(\mathrm{M}_1)_1 \text{ and } (\mathrm{T}_1)_1]}{1TS}\right)^1 \rightarrow \left(\frac{\mathrm{T}_{22a-1}[(\mathrm{M}_2,\ \mathrm{M}_3, \mathrm{M}_4)_1 \text{ and } (\mathrm{T}_{12})_1]}{1TS} \rightarrow \frac{\mathrm{T}_{22a-2}[(\mathrm{M}_2,\ \mathrm{M}_3, \mathrm{M}_4)_1 \text{ and } (\mathrm{T}_{12})_1]}{1TS}\right)^1$$

$$\rightarrow \left(\frac{\mathrm{T}_{23a-1}[(\mathrm{M}_2,\ \mathrm{M}_3, \mathrm{M}_4)_1 \text{ and } (\mathrm{T}_5, \mathrm{T}_6, \mathrm{T}_{11})_1]}{1TS} \rightarrow \frac{\mathrm{T}_{23a-2}[(\mathrm{M}_2,\ \mathrm{M}_3, \mathrm{M}_4)_1 \text{ and } (\mathrm{T}_6, \mathrm{T}_7, \mathrm{T}_{11})_1]}{1TS}\right)^1$$

Job #3: $O_{33} \rightarrow O_{31} \rightarrow O_{32}$

$$\left(\frac{\mathrm{T}_{31a-2}[(\mathrm{M}_2,\ \mathrm{M}_3, \mathrm{M}_4)_1 \text{ and } (\mathrm{T}_7, \mathrm{T}_8)_1]}{1TS}\right)^1 \rightarrow \left(\frac{\mathrm{T}_{32a-1}[(\mathrm{M}_2,\ \mathrm{M}_3, \mathrm{M}_4)_1 \text{ and } (\mathrm{T}_7, \mathrm{T}_8)_1]}{1TS} \rightarrow \frac{\mathrm{T}_{32a-2}[(\mathrm{M}_2,\ \mathrm{M}_3, \mathrm{M}_4)_1 \text{ and } (\mathrm{T}_7, \mathrm{T}_8)_1]}{1TS}\right)^1$$

$$\rightarrow \left(\frac{\mathrm{T}_{33a-1}[(\mathrm{M}_2,\ \mathrm{M}_3, \mathrm{M}_4)_1 \text{ and } (\mathrm{T}_7, \mathrm{T}_8)_1]}{1TS} \rightarrow \frac{\mathrm{T}_{33a-2}[(\mathrm{M}_2,\ \mathrm{M}_3, \mathrm{M}_4)_1 \text{ and } (\mathrm{T}_7, \mathrm{T}_8)_1]}{1TS}\right)^1$$

Job #4: $O_{42} \rightarrow O_{44} \rightarrow O_{41} \rightarrow O_{43}$

$$\left(\frac{\mathrm{T}_{41b-2}[(\mathrm{M}_3)_1 \text{ and } (\mathrm{T}_9, \mathrm{T}_{10})_1]}{1TS}\right)^1 \rightarrow \left(\frac{\mathrm{T}_{42a-1}[(\mathrm{M}_2, \mathrm{M}_3)_1 \text{ and } (\mathrm{T}_1, \mathrm{T}_3)_1]}{1TS} \rightarrow \frac{\mathrm{T}_{42a-2}[(\mathrm{M}_2, \mathrm{M}_3)_1 \text{ and } (\mathrm{T}_1, \mathrm{T}_3)_1]}{1TS} \rightarrow \frac{\mathrm{T}_{42a-3}[(\mathrm{M}_2, \mathrm{M}_3)_1 \text{ and } (\mathrm{T}_1, \mathrm{T}_3)_1]}{1TS}\right)^1$$

$$\rightarrow \left(\frac{\mathrm{T}_{43a-1}[(\mathrm{M}_2, \mathrm{M}_3)_1 \text{ and } (\mathrm{T}_6, \mathrm{T}_9)_1]}{1TS} \rightarrow \frac{\mathrm{T}_{43a-2}[(\mathrm{M}_2, \mathrm{M}_3)_1 \text{ and } (\mathrm{T}_6, \mathrm{T}_9)_1]}{1TS}\right)^1$$

$$\rightarrow \left(\begin{matrix} \frac{\mathrm{T}_{44a-1}[(\mathrm{M}_2)_1 \text{ and } (\mathrm{T}_3)_1]}{1TS} \rightarrow \frac{\mathrm{T}_{44a} - 2[(\mathrm{M}_2)_1 \text{ and } (\mathrm{T}_3)_1]}{1TS} \\ \frac{\mathrm{T}_{44b-1}[(\mathrm{M}_3)_1 \text{ and } (\mathrm{T}_3)_1]}{1TS} \rightarrow \frac{\mathrm{T}_{44b-2}[(\mathrm{M}_3)_1 \text{ and } (\mathrm{T}_3)_1]}{1TS} \rightarrow \frac{\mathrm{T}_{44b-3}[(\mathrm{M}_3)_1 \text{ and } (\mathrm{T}_3)_1]}{1TS} \end{matrix}\right)^1$$

Figure 4-10. Updated Remaining Tasks Information for the Following Time Slot

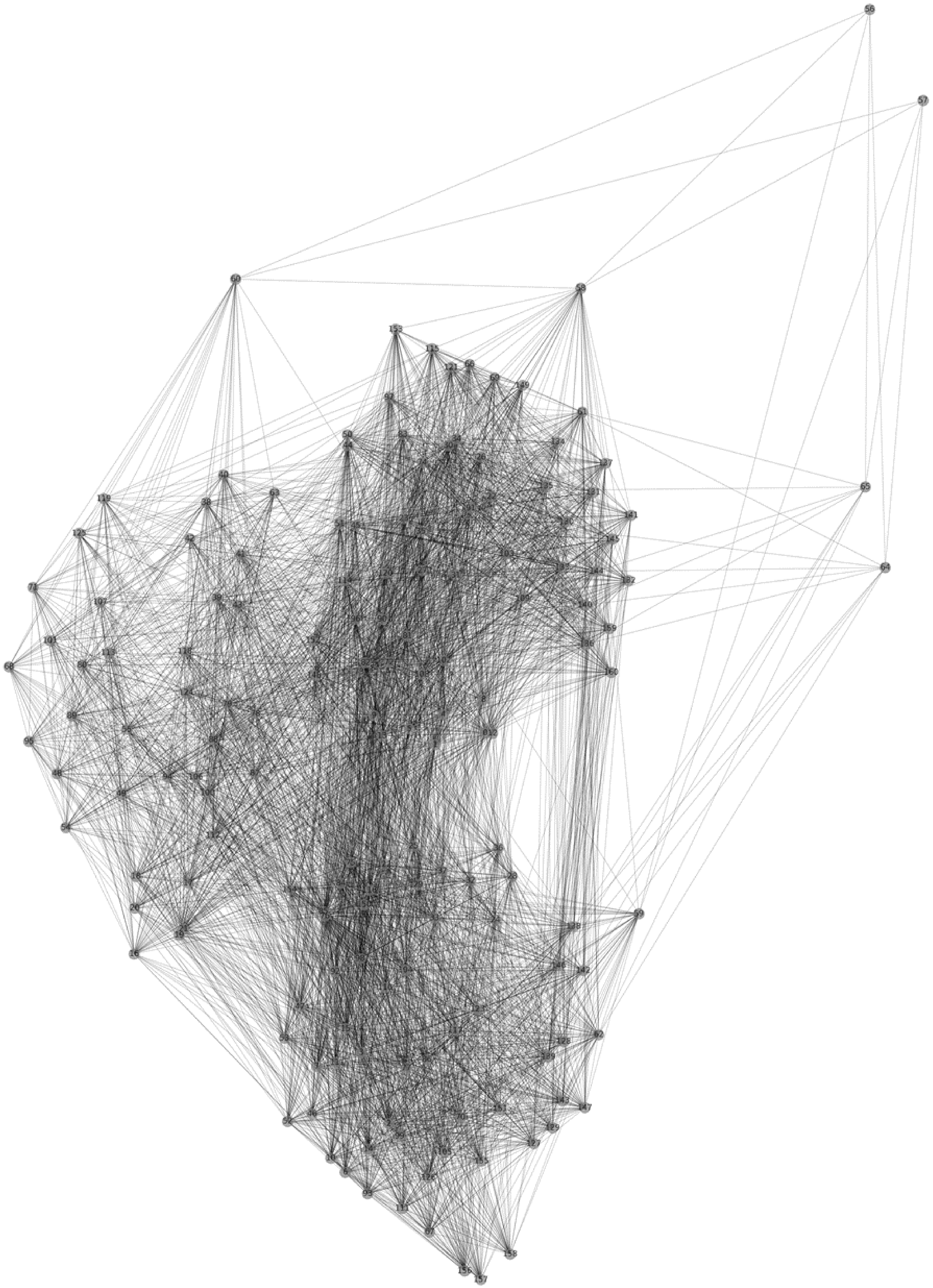
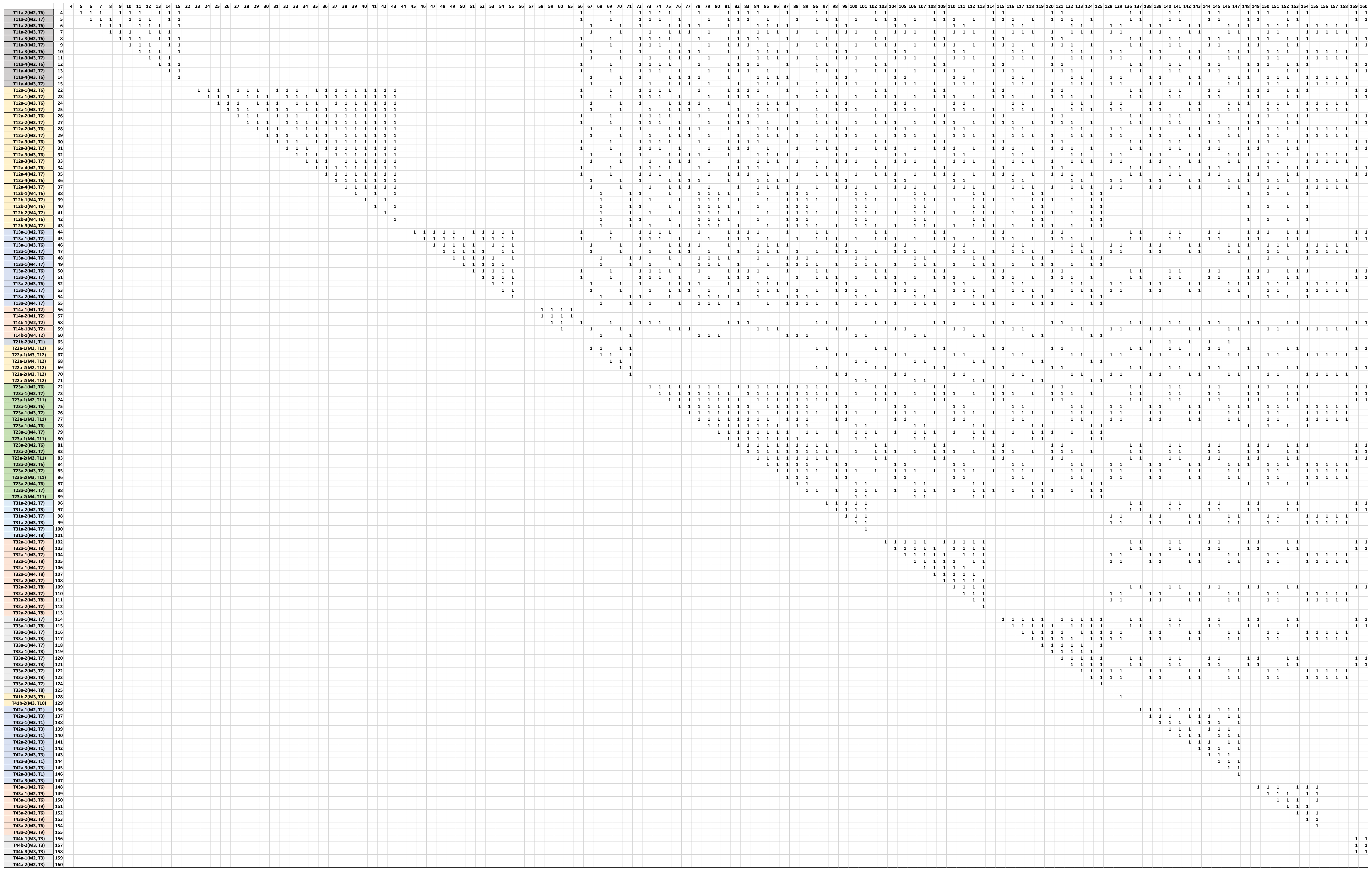


Figure 4-11. Updated Remaining Conflicting Graph for the Following Time Slot



T11a-2(M2, T6)	4
T11a-2(M2, T7)	5
T11a-2(M3, T6)	6
T11a-2(M3, T7)	7
T11a-3(M2, T6)	8
T11a-3(M2, T7)	9
T11a-3(M3, T6)	10
T11a-3(M3, T7)	11
T11a-4(M2, T6)	12
T11a-4(M2, T7)	13
T11a-4(M3, T6)	14
T11a-4(M3, T7)	15
T12a-1(M2, T6)	22
T12a-1(M2, T7)	23
T12a-1(M3, T6)	24
T12a-1(M3, T7)	25
T12a-2(M2, T6)	26
T12a-2(M2, T7)	27
T12a-2(M3, T6)	28
T12a-2(M3, T7)	29
T12a-3(M2, T6)	30
T12a-3(M2, T7)	31
T12a-3(M3, T6)	32
T12a-3(M3, T7)	33
T12a-4(M2, T6)	34
T12a-4(M2, T7)	35
T12a-4(M3, T6)	36
T12a-4(M3, T7)	37
T12b-1(M4, T6)	38
T12b-1(M4, T7)	39
T12b-2(M4, T6)	40
T12b-2(M4, T7)	41
T12b-3(M4, T6)	42
T12b-3(M4, T7)	43
T13a-1(M2, T6)	44
T13a-1(M2, T7)	45
T13a-1(M3, T6)	46
T13a-1(M3, T7)	47
T13a-1(M4, T6)	48
T13a-1(M4, T7)	49
T13a-2(M2, T6)	50
T13a-2(M2, T7)	51
T13a-2(M3, T6)	52
T13a-2(M3, T7)	53
T13a-2(M4, T6)	54
T13a-2(M4, T7)	55
T14a-1(M1, T2)	56
T14a-2(M1, T2)	57
T14b-1(M2, T2)	58
T14b-1(M3, T2)	59
T14b-1(M4, T2)	60
T21b-2(M1, T1)	65
T22a-1(M2, T12)	66
T22a-1(M3, T12)	67
T22a-1(M4, T12)	68
T22a-2(M2, T12)	69
T22a-2(M3, T12)	70
T22a-2(M4, T12)	71
T23a-1(M2, T6)	72
T23a-1(M2, T7)	73
T23a-1(M2, T11)	74
T23a-1(M3, T6)	75
T23a-1(M3, T7)	76
T23a-1(M3, T11)	77
T23a-1(M4, T6)	78
T23a-1(M4, T7)	79
T23a-1(M4, T11)	80
T23a-2(M2, T6)	81
T23a-2(M2, T7)	82
T23a-2(M2, T11)	83
T23a-2(M3, T6)	84
T23a-2(M3, T7)	85
T23a-2(M3, T11)	86
T23a-2(M4, T6)	87
T23a-2(M4, T7)	88
T23a-2(M4, T11)	89
T31a-2(M2, T7)	96
T31a-2(M2, T8)	97
T31a-2(M3, T7)	98
T31a-2(M3, T8)	99
T31a-2(M4, T7)	100
T31a-2(M4, T8)	101
T32a-1(M2, T7)	102
T32a-1(M2, T8)	103
T32a-1(M3, T7)	104
T32a-1(M3, T8)	105
T32a-1(M4, T7)	106
T32a-1(M4, T8)	107
T32a-2(M2, T7)	108
T32a-2(M2, T8)	109
T32a-2(M3, T7)	110
T32a-2(M3, T8)	111
T32a-2(M4, T7)	112
T32a-2(M4, T8)	113
T33a-1(M2, T7)	114
T33a-1(M2, T8)	115
T33a-1(M3, T7)	116
T33a-1(M3, T8)	117
T33a-1(M4, T7)	118
T33a-1(M4, T8)	119
T33a-2(M2, T7)	120
T33a-2(M2, T8)	121
T33a-2(M3, T7)	122
T33a-2(M3, T8)	123
T33a-2(M4, T7)	124
T33a-2(M4, T8)	125
T41b-2(M3, T9)	128
T41b-2(M3, T10)	129
T42a-1(M2, T1)	136
T42a-1(M2, T3)	137
T42a-1(M3, T1)	138
T42a-1(M2, T3)	139
T42a-2(M2, T1)	140
T42a-2(M2, T3)	141
T42a-2(M3, T1)	142
T42a-2(M3, T3)	143
T42a-3(M2, T1)	144
T42a-3(M2, T3)	145
T42a-3(M3, T1)	146
T42a-3(M3, T3)	147
T43a-1(M2, T6)	148
T43a-1(M2, T9)	149
T43a-1(M3, T6)	150
T43a-1(M3, T9)	151
T43a-2(M2, T6)	152
T43a-2(M2, T9)	153
T43a-2(M3, T6)	154
T43a-2(M3, T9)	155
T44b-1(M3, T3)	156
T44b-2(M3, T3)	157
T44b-3(M3, T3)	158
T44a-1(M2, T3)	159
T44a-2(M2, T3)	160

Figure 4-12. Updated Remaining Edge Connection Details for the Following Time Slot

Step #5: Checking the ending condition

In step five, we need to make a judgment. If there is at least one remaining Unit Task, go to step #3. If there is no remaining Unit Task, the PPS problem computation is finished, and the output schedule is the combination of Unit Task Candidates and Candidate Nodes found at each time slot.

The results of the example problem with Heuristics #13 is illustrated in Figure 4-13. Our approach can get to a near-optimal solution finishing in 107.5 time unit compare to the optimum solution finishing in 98 time units, which is a 9.69% error. The computation of our approach takes about 20 seconds, which is much faster (seconds versus days) compare to the optimum solutions using integer programming.

Table 4-5 shows the performance of our approach in terms of accuracy and computation time. Each column describes the heuristics ID, the minimum makespan in a number of time slots, average clock time in 3-run, whether it is an approximation algorithm, error in a number of time slots, and error rate, respectively. The accuracy is the error rate by comparing the result of my approach with the optimal solution, and computation time is the clock time taken to finish the computation. For the example problem, we can get a near-optimal in seconds with a minimum 10% error but much faster. In the following chapter, we test our approach on a real-world example from the literature, and further test cases are designed to exam the accuracy, robustness, and scalability of our approach.

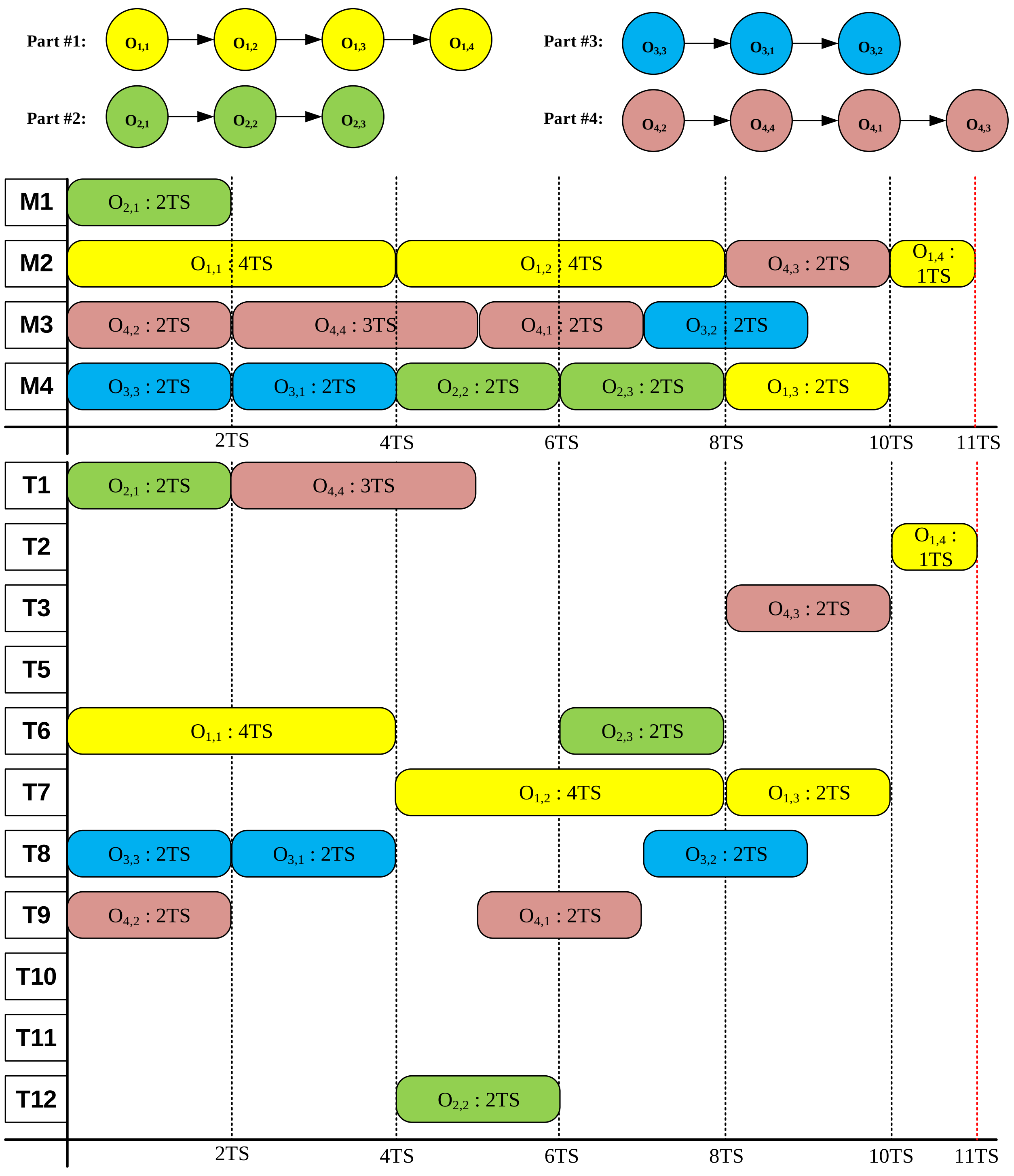


Figure 4-13. Schedule Created with Heuristics #13

Table 4-5. Outputs of the Heuristics Configurations on the Example PPS Problem

Methods	**Minimum makespan (in time slots)**	**Clock time, 3-run average**	**Is approx?**	**Error**	**Error rate**
Heuristics#1	11	11768.55	No	1	10%
Heuristics#2	11	14392.62	No	1	10%
Heuristics#3	11	13370.07	No	1	10%
Heuristics#4	11	13344.73	No	1	10%
Heuristics#5	11	10904.36	No	1	10%
Heuristics#6	11	12042.11	No	1	10%
Heuristics#7	14	11942.88	No	4	40%
Heuristics#8	11	10178.87	No	1	10%
Heuristics#9	11	10496.09	No	1	10%
Heuristics#10	11	10833.84	No	1	10%
Heuristics#11	12	8.75	Yes	2	20%
Heuristics#12	11	38.73	Yes	1	10%
Heuristics#13	11	26.02	Yes	1	10%
Heuristics#14	11	9.23	Yes	1	10%
Heuristics#15	11	39.22	Yes	1	10%
Heuristics#16	11	28.09	Yes	1	10%
Heuristics#17	11	9.16	Yes	1	10%
Heuristics#18	11	37.66	Yes	1	10%
Heuristics#19	11	26.74	Yes	1	10%
Heuristics#20	14	9.99	Yes	4	40%
Heuristics#21	11	43.34	Yes	1	10%
Heuristics#22	11	27.02	Yes	1	10%
Heuristics#23	14	9.78	Yes	4	40%
Heuristics#24	11	41.62	Yes	1	10%
Heuristics#25	11	27.59	Yes	1	10%
Heuristics#26	14	9.91	Yes	4	40%
Heuristics#27	11	40.42	Yes	1	10%
Heuristics#28	11	28.64	Yes	1	10%

4.6 Summary

In this chapter, we propose a novel approach to formulate a general type of PPS problem with integrated resource allocation and process planning towards a typical objective, minimizing the makespan. The PPS problem is formulated into an undirect weighted conflicting graph due to its nature of resource and sequence constraints. In this conflicting graph, nodes stand for operations and their resources; edges stand for constraints, and weight factors are the guidelines for the node selection at each time slot. A variation of the GCP, the MWIS problem, can be solved to find the best set of operations with their desired resources at each discrete time slot.

This proposed approach solves the PPS problem directly with minimum iteration. We establish that the proposed approach always returns a feasible solution by selecting the MWIS for each time slot. The accuracy and computational speed of the MWIS algorithm in the heuristics configurations can guarantee the performance of the proposed approach.

The seven weight factor arrangements, together with the eight MWIS algorithms from Chapter 3, are constructed into twenty-eight heuristics configurations for solving the PPS problem. These heuristics configurations are listed in Table 4-3. In the following Chapter 5, we test our approach on a real-world example from the literature, and further test instances are designed to exam the accuracy, robustness, and scalability of our approach.

Chapter 5. Computational Experiments

*In this chapter, we first want to verify the feasibility of the proposed approach for the **P**rocess **P**lanning and **S**cheduling (PPS) problem on a real-world example. Secondly, we create a set of testing cases based on the structure of the real-world example but randomized sequencing constraints and resource combinations for further evaluation. The results obtained on all the test instances are reported and analyzed in terms of scalability, accuracy, and robustness. The scalability analysis shows how the proposed approach behaves on different sizes of the inputs. The accuracy refers to how likely the proposed approach can get to the optimum results. It can be measured by the average and maximum error rate on the tests. And the robustness ensures the error-free and bug-free on all the tests.*

*Chapter 5 is organized as following sections: Section 5.1 provides the details of the implementation of the **I**nteger **P**rogramming (IP) model for the PPS problem, so that the optimum solutions can be obtained. Section 5.2 describes the proposed approach with a real-world problem from the literature. Section 5.3 discusses and analyzes the results of all the test instances. Section 5.4 gives the summary of Chapter 5.*

*5.1 Integer Programming Model for **P**rocess **P**lanning and **S**cheduling (PPS) Problem*

5.1.1 Implementation of **I**nteger **P**rogramming (IP) Model

In order to get the optimum solution to the PPS problem, the **I**nteger **P**rogramming (IP) model is implemented and tested based on the mathematical modeling discussed in section 4.3. The IP model is implemented with python package “pyomo” in Python 3.7.5. The solver utilized in this implementation is “glpk (GNU Linear Programming Kit).”

Part #1: $\frac{T_{1,1}[(R_1,R_2,R_3)_2]}{2} \rightarrow \frac{T_{1,2}[(R_1,R_2,R_3)_2]}{1} \rightarrow \frac{T_{1,3}[(R_1,R_2,R_3)_1]}{2} \rightarrow \frac{T_{1,4}[(R_4)_1]}{1}$

Part #2: $\frac{T_{2,1}[(R_1,R_2,R_3)_1]}{1} \rightarrow \frac{T_{2,2}[(R_1)_1 and (R_2,R_3)_1]}{2} \rightarrow \frac{T_{2,3}[(R_4)_1]}{1}$

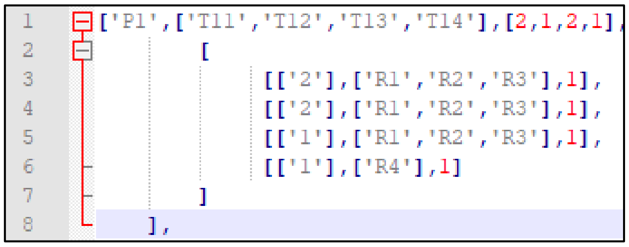
Part #3: $\frac{T_{3,1}[(R_1,R_2)_1]}{1} \rightarrow \begin{pmatrix} \frac{T_{3,2a}[(R_1,R_2)_2]}{1} \\ \frac{T_{3,2b}[(R_3)_1]}{2} \end{pmatrix}^1 \rightarrow \frac{T_{3,3}[(R_3,R_4)_1]}{3}$

Part #4: $\frac{T_{4,1}[(R_1,R_2,R_3)_3]}{1} \rightarrow \frac{T_{4,2}[(R_1,R_2,R_3)_2]}{2} \rightarrow \frac{T_{4,3}[(R_1,R_2,R_3)_3]}{1} \rightarrow \frac{T_{4,4}[(R_1,R_2,R_3)_2]}{1} \rightarrow \frac{T_{4,5}[(R_4)_1]}{1}$

Where, R_1, R_2, R_3, R_4 are the four different resources. For $\frac{T_{1,1}[(R_1,R_2,R_3)_2]}{2}$, it means that the first operation (task $T_{1,1}$) of Part #1 requires any combination of two resources among (R_1, R_2, R_3) and the duration is 2 time slots. $\begin{pmatrix} \frac{T_{3,2a}[(R_1,R_2)_2]}{1} \\ \frac{T_{3,2b}[(R_3)_1]}{2} \end{pmatrix}^1$ means that the task $T_{3,2}$ can be processed with two task options $T_{3,2a}$ and $T_{3,2b}$.

Figure 5-1. Parts Information with Simplified Duration Information

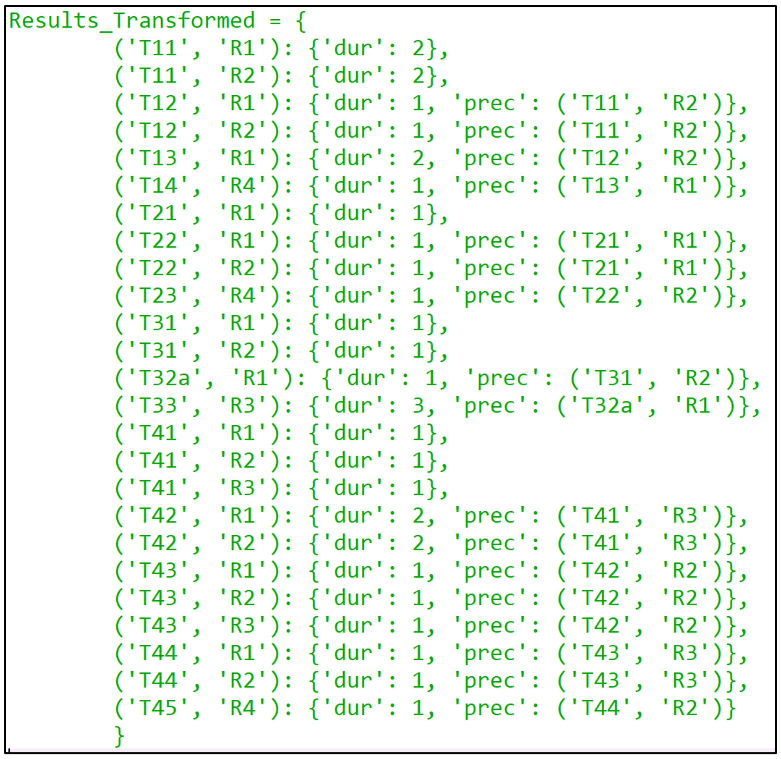
Assume that the best practice sequence of operations of four parts is given as Figure 5-1. Note that, we generalize the machines, tools and all other possible resources as r number of resources, $(R_1, R_2, \dots, R_r)$. The input format, taking Part #1 operations as an example, is shown in Figure 5-2.



```
['P1',['T11','T12','T13','T14'],[2,1,2,1],
        [
            [['2'],['R1','R2','R3'],1],
            [['2'],['R1','R2','R3'],1],
            [['1'],['R1','R2','R3'],1],
            [['1'],['R4'],1]
        ]
    ],
```

Figure 5-2. Input of Part #1 Operations

The inputs are then transformed into the dictionary shown as Figure 5-3 to fulfill the solver's requirements. Each job is broken down into task-resource pairs associated with its duration and sequencing information. For tasks that require more than one resource, each required resource is generated as one task-resource pair instance.



```
Results_Transformed = {
        ('T11', 'R1'): {'dur': 2},
        ('T11', 'R2'): {'dur': 2},
        ('T12', 'R1'): {'dur': 1, 'prec': ('T11', 'R2')},
        ('T12', 'R2'): {'dur': 1, 'prec': ('T11', 'R2')},
        ('T13', 'R1'): {'dur': 2, 'prec': ('T12', 'R2')},
        ('T14', 'R4'): {'dur': 1, 'prec': ('T13', 'R1')},
        ('T21', 'R1'): {'dur': 1},
        ('T22', 'R1'): {'dur': 1, 'prec': ('T21', 'R1')},
        ('T22', 'R2'): {'dur': 1, 'prec': ('T21', 'R1')},
        ('T23', 'R4'): {'dur': 1, 'prec': ('T22', 'R2')},
        ('T31', 'R1'): {'dur': 1},
        ('T31', 'R2'): {'dur': 1},
        ('T32a', 'R1'): {'dur': 1, 'prec': ('T31', 'R2')},
        ('T33', 'R3'): {'dur': 3, 'prec': ('T32a', 'R1')},
        ('T41', 'R1'): {'dur': 1},
        ('T41', 'R2'): {'dur': 1},
        ('T41', 'R3'): {'dur': 1},
        ('T42', 'R1'): {'dur': 2, 'prec': ('T41', 'R3')},
        ('T42', 'R2'): {'dur': 2, 'prec': ('T41', 'R3')},
        ('T43', 'R1'): {'dur': 1, 'prec': ('T42', 'R2')},
        ('T43', 'R2'): {'dur': 1, 'prec': ('T42', 'R2')},
        ('T43', 'R3'): {'dur': 1, 'prec': ('T42', 'R2')},
        ('T44', 'R1'): {'dur': 1, 'prec': ('T43', 'R3')},
        ('T44', 'R2'): {'dur': 1, 'prec': ('T43', 'R3')},
        ('T45', 'R4'): {'dur': 1, 'prec': ('T44', 'R2')}
        }
```

Figure 5-3. Inputs Dictionary Format for Package "pyomo" in Python

The mathematical modeling of PPS problem from section 4.3 is transformed into the format for the python package "pyomo" as well as the solver "glpk". The new formulation is as below:

(1) The variables

- model.start = pyo.Var(PARTS, RESOURCES, domain = pyo.NonNegativeReals)
- model.makespan = pyo.Var(domain=pyo.NonNegativeReals)
- model.y = pyo.Var(PARTS,PARTS,RESOURCES, domain = pyo.Boolean)

(2) The objective

- model.Obj = pyo.Objective(expr = model.makespan, sense = pyo.minimize)

(3) The constraints

For the instances of the same tasks but different resources, these instances must have the same start time.

- model.cons.add(model.start[j,r] <= model.start[m,n])
- model.cons.add(model.start[m,n] <= model.start[j,r])

The makespan is the finishing time for all tasks.

- model.cons.add(model.start[j,m] + TASKS[(j,m)]['dur'] <= model.makespan)

For a task which requires a predecessor, it can only be scheduled after the predecessor is finished.

- model.cons.add(model.start[j,m] >= model.start[k,n] + TASKS[(k,n)]['dur'])

For the tasks who shares resources, they cannot be scheduled in the same time.

- model.cons.add(model.start[j,m] + TASKS[(j,m)]['dur'] <= model.start[k,m]

or

model.cons.add(model.start[k,m] + TASKS[(k,m)]['dur'] <= model.start[j,m]

5.1.2 Numerical Results of IP Model

We introduce the **I**nput **C**omplexity **I**ndex (ICI) to measure the complexity of inputs. It is essentially a reference value describing the relative size of the possible number of combinations of results of the PPS problem. As discussed in Chapter 4, the PPS problem can be understood as a conflicting graph so that we can utilize some parameters of this graph to calculate the ICI. The ICI can be defined as,

$$ICI = |P|^{(\frac{|T|}{|P|}-1)} * |N| * |O| * |D|$$

Where, $|P|$ is the number of parts; $|T|$ is the number of tasks; $|O|$ is the total number of the options of the tasks. $|D|$ is the density of the conflicting graph, and $|N|$ is the number of nodes of the conflicting graph.

The graph density is defined as follows (Diestel, 2006),

$$D = \frac{2|E|}{|N|(|N|-1)}$$

Where $|E|$ is the number of edges in the conflicting graph.

IP is NP-complete on discrete problems, which means that its computation time should increase exponentially with the size of the inputs. To verify this hypothesis, we simulated 10 PPS problems considering a different number of parts and operations, as well as diverse information for operations. The results are shown in Table 5-1. In Table 5-1, each column describes the test ID, number of parts, number of tasks, number of edges, number of nods, graph density, number of options, ICI, 3-run average clock time in seconds, minimum makespan, respectively. Note that all computational experiments in this thesis are performed on a virtual server at Syracuse University. The CPU is Intel Xeon E5-2699 with a fixed maximum speed at 2.3 GHz, and the memory is 32 GB. All the implementations mentioned in this thesis are in single threading.

Table 5-1. Integer Programming Model Numerical Results

Test -ID	# of Parts	# of Tasks	# of Edges	# of Nodes	Graph Density	# of options	Input Complexity Index	Clock Time (s)	Minimum Makespan
1	2	7	111	24	0.40	1	218.40	6.59	7
2	3	10	227	35	0.38	2	3119.84	68.67	8
3	3	11	307	37	0.46	1	2873.64	26.97	16
4	4	12	315	41	0.38	2	8064	118.94	11
5	4	12	390	41	0.48	2	9984	150.77	10

6	5	13	316	40	0.41	2	10640.80	643.77	7
7	5	14	396	41	0.48	2	17938.30	5794.50	10
8	5	14	504	45	0.51	2	20755.05	12196.93	10
9	4	14	375	41	0.46	1	9600	213.92	18
10	4	14	1074	63	0.55	1	17738.32	2959.31	18

The computation time follows an exponential trendline with increasing input ICI in Figure 5-4, and the logarithmic computation time follows a straight trendline with increasing input ICI in Figure 5-5. It can be seen that the IP model follows an exponential complexity of the PPS problem. Although the solution of the IP model can provide the optimum solution to the PPS problem, the computational speed is unacceptable. But we can manipulate inputs based on the outputs of our approach, so that the outputs of our approach can be verified in terms of accuracy.

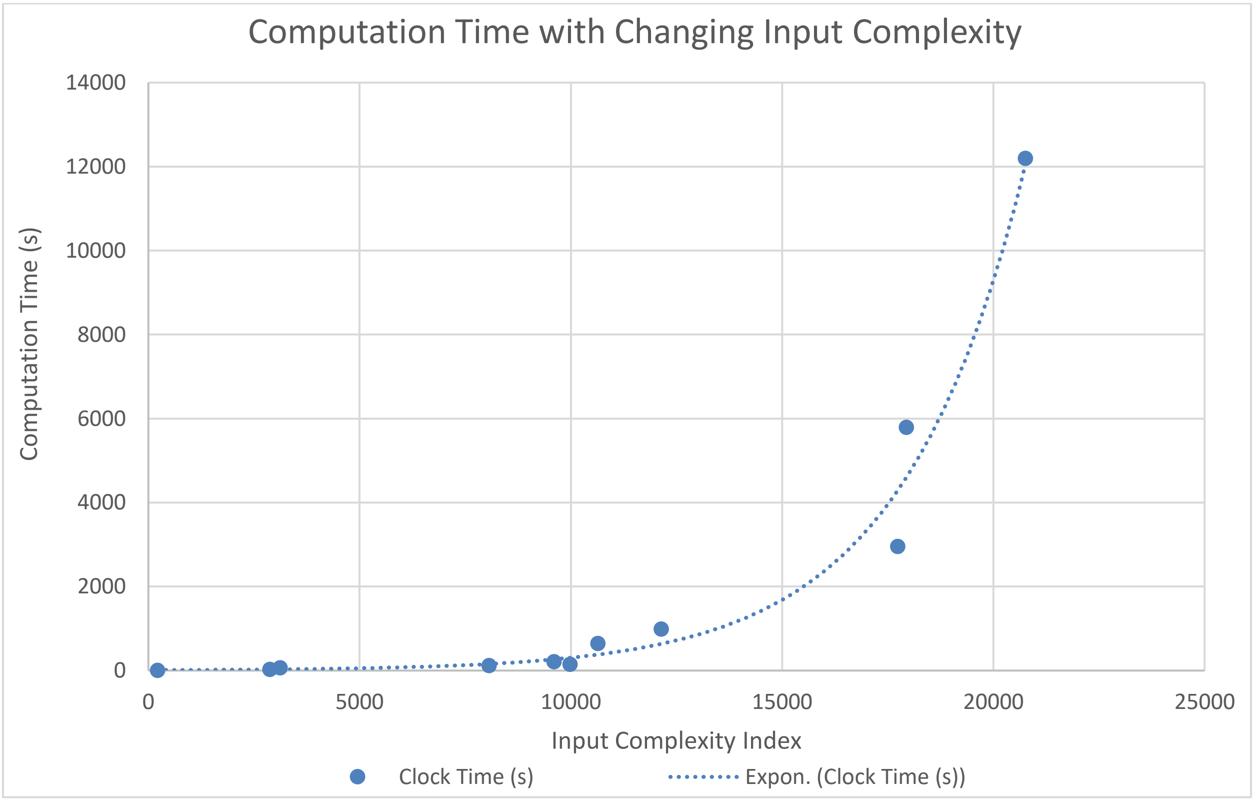


Figure 5-4. Computation Time with Changing ICI

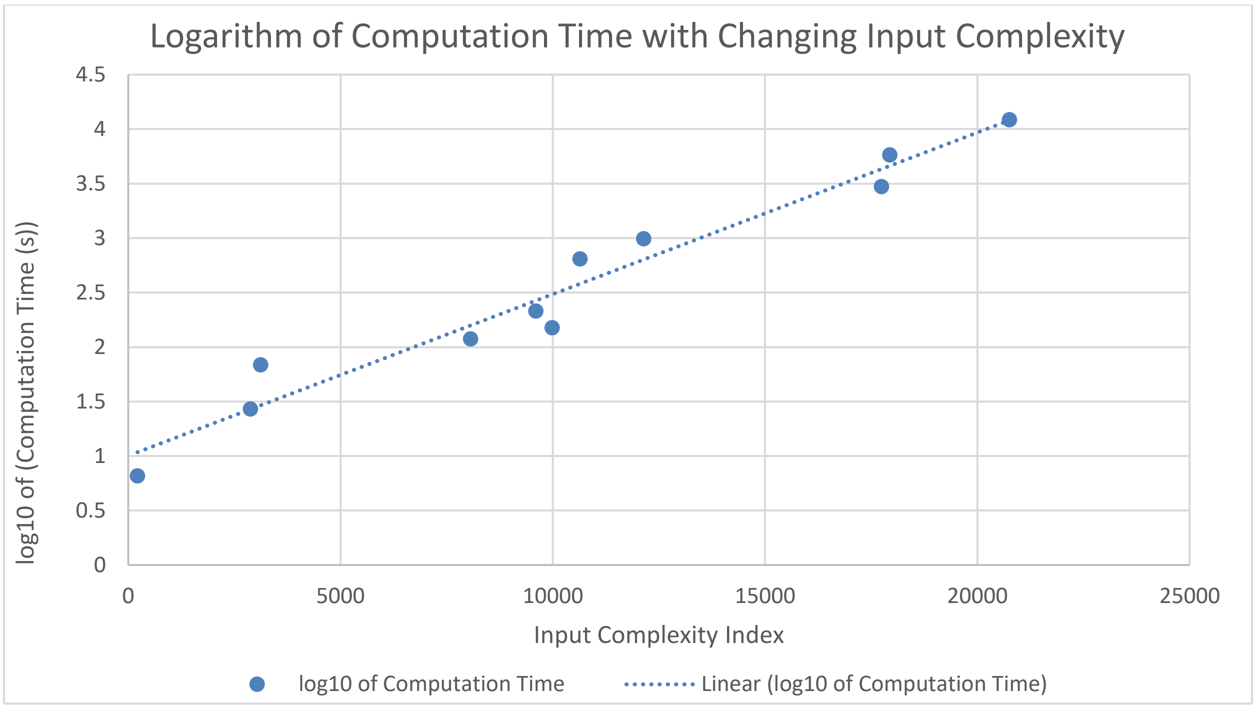


Figure 5-5. Logarithmic Computation Time with Changing ICI

5.2 A Real-world Example Using the Proposed Approach

Based on the case study from Zhang et al.'s work (Zhang et al., 2014) and combined with the details from Zhang et al.'s references (Chu & Gadh, 1996; Zhang et al., 2003; Li et al., 2005; Li & McMahon, 2007), we constructed a real-world PPS problem to verify our approach. The resources, machines, and cutting tools of a specific job shop are defined in Table 5-2. The four parts of the problem are shown in Figure 5-6. The relevant technical specifications of the four parts are defined in Tables 5-3 to 5-6.

Table 5-2. The Resource of a Job Shop – Machines and Tools

Types	**No.**
Machines	
Drilling press	M_1
Three-axis vertical milling machine I	M_2
Three-axis vertical milling machine II	M_3
CNC three-axis vertical milling machine	M_4

Boring machine	M_5
Cutting tools	
Drill 1	T_1
Drill 2	T_2
Drill 3	T_3
Drill 4	T_4
Tapping tool	T_5
Mill 1	T_6
Mill 2	T_7
Mill 3	T_8
Reaming tool	T_9
Boring tool	T_{10}
Slot cutter	T_{11}
Chamfer tool	T_{12}

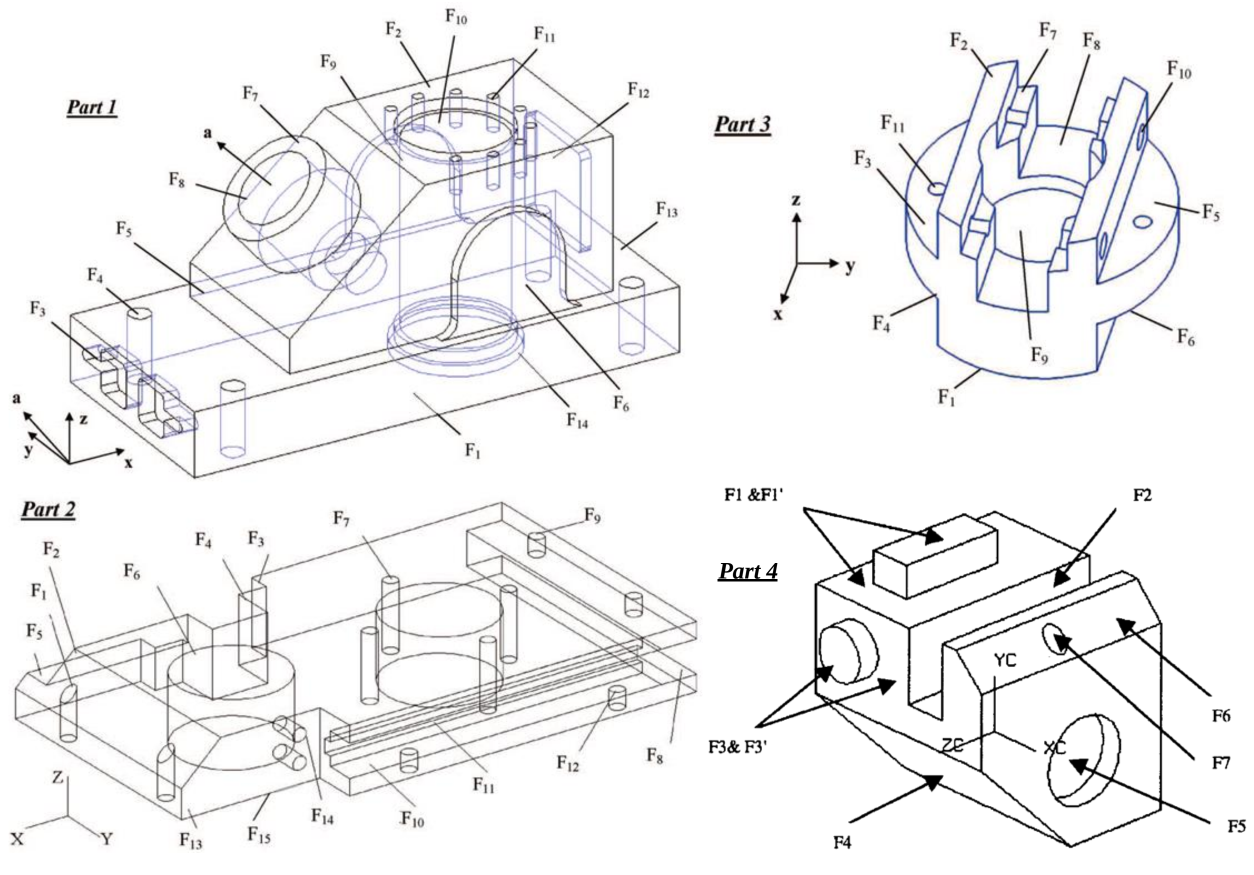


Figure 5-6. The description of 4 parts of PPS

Table 5-3. The Technical Specifications for Part #1

Features	**Operations**	**Machine Candidates**	**Tool Candidates**	**Machining time for each candidate machine (s)**
F1	Milling (Oper1)	M_2, M_3, M_4	T_6, T_7, T_8	40, 40, 30
F2	Milling (Oper2)	M_2, M_3, M_4	T_6, T_7, T_8	40, 40, 30
F3	Milling (Oper3)	M_2, M_3, M_4	T_6, T_7, T_8	20, 20, 15
F4	Drilling (Oper4)	M_1, M_2, M_3, M_4	T_2	12, 10, 10, 7.5
F5	Milling (Oper5)	M_2, M_3, M_4	T_6, T_7	35, 35, 26.25
F6	Milling (Oper6)	M_2, M_3, M_4	T_7, T_8	15, 15, 11.25
F7	Milling (Oper7)	M_2, M_3, M_4	T_7, T_8	30, 30, 22.5
F8	Milling (Oper8)	M_1, M_2, M_3, M_4	T_2, T_3, T_4	21.6, 18, 18, 13.5
	Reaming (Oper9)	M_2, M_3, M_4	T_9	10, 10, 7.5
	Boring (Oper10)	M_2, M_3, M_4, M_5	T_{10}	10, 10, 7.5, 12
F9	Milling (Oper11)	M_2, M_3, M_4	T_7, T_8	15, 15, 11.25
F10	Drilling (Oper12)	M_1, M_2, M_3, M_4	T_2, T_3, T_4	48, 40, 40, 30
	Reaming (Oper13)	M_2, M_3, M_4	T_9	25, 25, 18.75
	Boring (Oper14)	M_2, M_3, M_4, M_5	T_{10}	25, 25, 18.75, 30
F11	Milling (Oper15)	M_1, M_2, M_3, M_4	T_1	26.4, 22, 22, 16.5
	Tapping (Oper16)	M_2, M_3, M_4	T_5	20, 20, 15
F12	Milling (Oper17)	M_2, M_3, M_4	T_7, T_8	16, 16, 12
F13	Milling (Oper18)	M_2, M_3, M_4	T_6, T_7	35, 35, 26.25
F14	Reaming (Oper19)	M_2, M_3, M_4	T_9	12, 12, 9
	Boring (Oper20)	M_2, M_3, M_4, M_5	T_{10}	12, 12, 9, 14.4

Table 5-4. The Technical Specifications for Part #2

Features	**Operations**	**Machine Candidates**	**Tool Candidates**	**Machining time for each candidate machine (s)**
F1	Drilling (Oper1)	M_1, M_2, M_3, M_4	T_1	12, 10, 10, 7.5
F2	Milling (Oper2)	M_2, M_3, M_4	T_{12}	20, 20, 15
F3	Milling (Oper3)	M_2, M_3, M_4	T_5, T_6, T_{11}	18, 18, 13.5
F4	Milling (Oper4)	M_2, M_3, M_4	T_6, T_7, T_8	16, 16, 12
F5	Milling (Oper5)	M_2, M_3, M_4	T_6, T_7, T_8	15, 15, 11.25
F6	Drilling (Oper6)	M_1, M_2, M_3, M_4	T_2	30, 25, 25, 18.75
	Reaming (Oper7)	M_2, M_3, M_4	T_9	25, 25, 18.75
F7	Drilling (Oper8)	M_1, M_2, M_3, M_4	T_1	14.4, 12, 12, 9
F8	Milling (Oper9)	M_2, M_3, M_4	T_6, T_7, T_8	15, 15, 11.25
F9	Drilling (Oper10)	M_1, M_2, M_3, M_4	T_1	9.6, 8, 8, 6
F10	Milling (Oper11)	M_2, M_3, M_4	T_6, T_7, T_8	10, 10, 7.5
F11	Milling (Oper12)	M_2, M_3, M_4	T_6, T_7, T_8	10, 10, 7.5
F12	Drilling (Oper13)	M_1, M_2, M_3, M_4	T_1	9.6, 8, 8, 6
F13	Milling (Oper14)	M_2, M_3, M_4	T_6, T_7, T_8	16, 16, 12
F14	Drilling (Oper15)	M_1, M_2, M_3, M_4	T_1	9.6, 8, 8, 6
F15	Milling (Oper16)	M_1, M_2, M_3, M_4	T_6, T_7, T_8	36, 30, 30, 22.5

Table 5-5. The Technical Specifications for Part #3

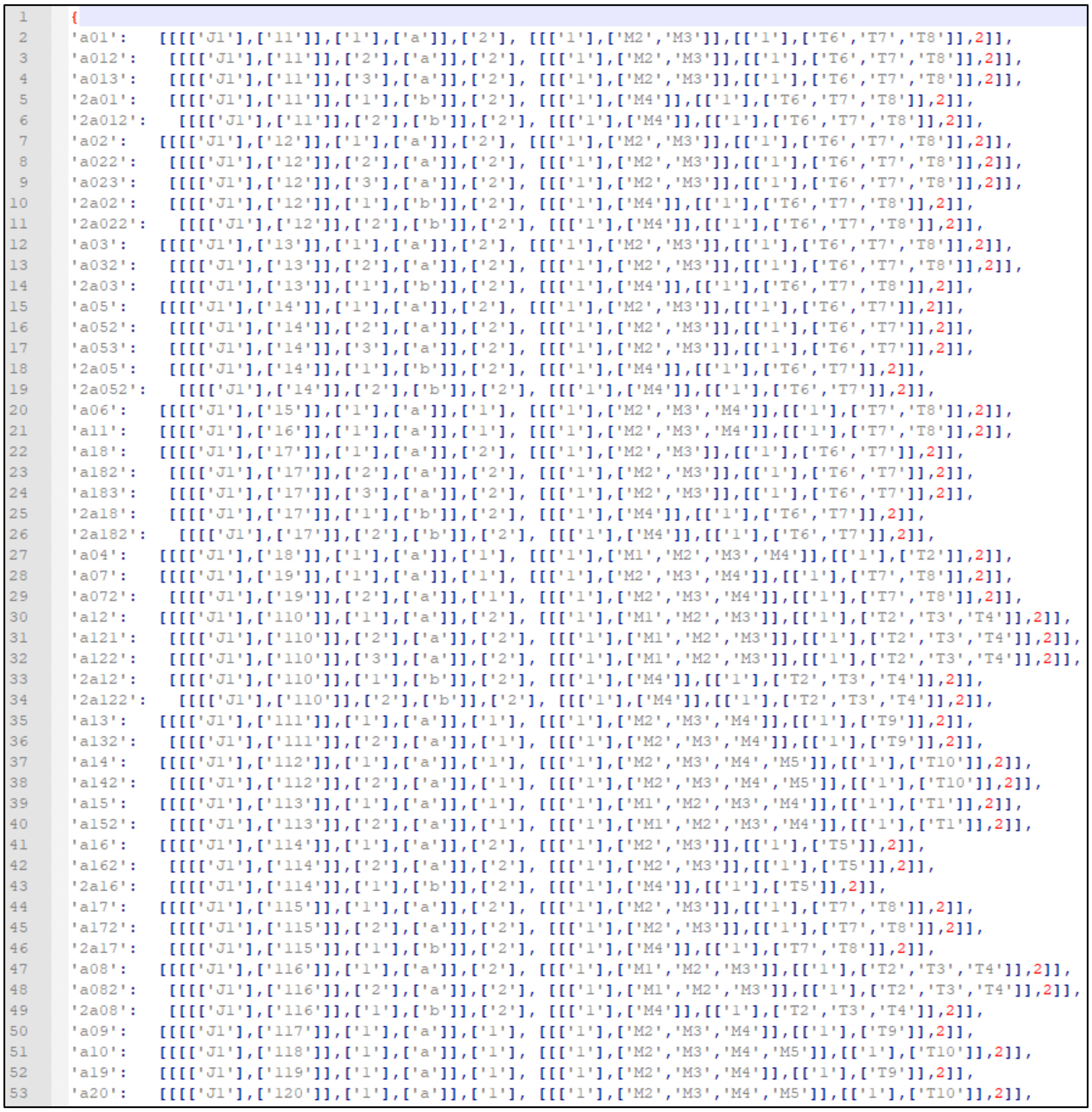
Features	Operations	Machine Candidates	Tool Candidates	Machining time for each candidate machine (s)
F1	Milling (Oper1)	M_2, M_3, M_4	T_6, T_7, T_8	20, 15, 20
F2	Milling (Oper2)	M_2, M_3, M_4	T_6, T_7, T_8	20, 15, 20
F3	Milling (Oper3)	M_2, M_3, M_4	T_6, T_7, T_8	15, 15, 11.25
F4	Milling (Oper4)	M_1, M_2, M_3, M_4	T_2	15, 15, 11.25, 18
F5	Milling (Oper5)	M_2, M_3, M_4	T_6, T_7, T_8	15, 15, 11.25
F6	Milling (Oper6)	M_2, M_3, M_4	T_7, T_8	15, 15, 11.25
F7	Milling (Oper7)	M_2, M_3, M_4	T_7, T_8, T_{11}	15, 15, 11.25
F8	Milling (Oper8)	M_2, M_3, M_4	T_6, T_7, T_8, T_{11}	25, 25, 18.75
F9	Drilling (Oper9)	M_1, M_2, M_3, M_4	T_2, T_3, T_4	30, 25, 25, 18.75
	Reaming (Oper10)	M_2, M_3, M_4	T_9	20, 20, 15
	Boring (Oper11)	M_2, M_3, M_4, M_5	T_{10}	20, 20, 15, 24
F10	Drilling (Oper12)	M_1, M_2, M_3, M_4	T_1	9.6, 8, 8, 6
	Tapping (Oper13)	M_2, M_3, M_4	T_5	8, 8, 6
F11	Drilling (Oper14)	M_1, M_2, M_3, M_4	T_9	6, 5, 5, 3.75

Table 5-6. The Technical Specifications for Part #4

Features	Operations	Machine Candidates	Tool Candidates	Machining time for each candidate machine (s)
F1	Milling (Oper1)	M_2, M_4	T_6, T_9	12
F2	Milling (Oper2)	M_2, M_4	T_9, T_{10}	21
F3	Milling (Oper3)	M_2, M_4	T_9	18
F4	Milling (Oper4)	M_2, M_4	T_1, T_9	27
F5	Drilling (Oper5)	M_1, M_2, M_4	T_2	20
F6	Milling (Oper6)	M_2, M_4	T_1, T_9	18
F7	Drilling (Oper7)	M_1, M_2, M_4	T_2	20

We define each time slot (1TS) representing 15 time units. The top segment of Figure 5-8 illustrates the best practice operation sequence of the four parts. All the operations are then transformed into the input format. Figure 5-7 shows the transformed operations of Part #1. There are 119 Unit Tasks for all the four parts. We can generate the conflicting graph, which has 47525 edges and 580 nodes. For a problem in such a size, the heuristics configurations with faster approximation-based algorithms are preferred.

Using the Heuristics #19, Algorithm A5 MWIS_SubCS_GWMIN and MWIS A3: MWIS Weights 3, and using the median length weight factor, $LW_c^M = 1$. The schedule with the resource allocations generated is shown in Figure 5-8. Table 5-7 shows the outputs of Heuristics #11 to Heuristics #28 on the real-world PPS problem. Among all the Heuristics tested, the Heuristics #19 achieved the optimum solution with 31 time slots. The results with an error rate of less than 5% take 7000~11000 seconds of clock time for finishing the computation.



```
{
'a01':   [[[['J1'],['11']],['1'],['a']],['2'], [[['1'],['M2','M3']],[['1'],['T6','T7','T8']],2]],
'a012':   [[[['J1'],['11']],['2'],['a']],['2'], [[['1'],['M2','M3']],[['1'],['T6','T7','T8']],2]],
'a013':   [[[['J1'],['11']],['3'],['a']],['2'], [[['1'],['M2','M3']],[['1'],['T6','T7','T8']],2]],
'2a01':   [[[['J1'],['11']],['1'],['b']],['2'], [[['1'],['M4']],[['1'],['T6','T7','T8']],2]],
'2a012':   [[[['J1'],['11']],['2'],['b']],['2'], [[['1'],['M4']],[['1'],['T6','T7','T8']],2]],
'a02':   [[[['J1'],['12']],['1'],['a']],['2'], [[['1'],['M2','M3']],[['1'],['T6','T7','T8']],2]],
'a022':   [[[['J1'],['12']],['2'],['a']],['2'], [[['1'],['M2','M3']],[['1'],['T6','T7','T8']],2]],
'a023':   [[[['J1'],['12']],['3'],['a']],['2'], [[['1'],['M2','M3']],[['1'],['T6','T7','T8']],2]],
'2a02':   [[[['J1'],['12']],['1'],['b']],['2'], [[['1'],['M4']],[['1'],['T6','T7','T8']],2]],
'2a022':   [[[['J1'],['12']],['2'],['b']],['2'], [[['1'],['M4']],[['1'],['T6','T7','T8']],2]],
'a03':   [[[['J1'],['13']],['1'],['a']],['2'], [[['1'],['M2','M3']],[['1'],['T6','T7','T8']],2]],
'a032':   [[[['J1'],['13']],['2'],['a']],['2'], [[['1'],['M2','M3']],[['1'],['T6','T7','T8']],2]],
'2a03':   [[[['J1'],['13']],['1'],['b']],['2'], [[['1'],['M4']],[['1'],['T6','T7','T8']],2]],
'a05':   [[[['J1'],['14']],['1'],['a']],['2'], [[['1'],['M2','M3']],[['1'],['T6','T7']],2]],
'a052':   [[[['J1'],['14']],['2'],['a']],['2'], [[['1'],['M2','M3']],[['1'],['T6','T7']],2]],
'a053':   [[[['J1'],['14']],['3'],['a']],['2'], [[['1'],['M2','M3']],[['1'],['T6','T7']],2]],
'2a05':   [[[['J1'],['14']],['1'],['b']],['2'], [[['1'],['M4']],[['1'],['T6','T7']],2]],
'2a052':   [[[['J1'],['14']],['2'],['b']],['2'], [[['1'],['M4']],[['1'],['T6','T7']],2]],
'a06':   [[[['J1'],['15']],['1'],['a']],['1'], [[['1'],['M2','M3','M4']],[['1'],['T7','T8']],2]],
'a11':   [[[['J1'],['16']],['1'],['a']],['1'], [[['1'],['M2','M3','M4']],[['1'],['T7','T8']],2]],
'a18':   [[[['J1'],['17']],['1'],['a']],['2'], [[['1'],['M2','M3']],[['1'],['T6','T7']],2]],
'a182':   [[[['J1'],['17']],['2'],['a']],['2'], [[['1'],['M2','M3']],[['1'],['T6','T7']],2]],
'a183':   [[[['J1'],['17']],['3'],['a']],['2'], [[['1'],['M2','M3']],[['1'],['T6','T7']],2]],
'2a18':   [[[['J1'],['17']],['1'],['b']],['2'], [[['1'],['M4']],[['1'],['T6','T7']],2]],
'2a182':   [[[['J1'],['17']],['2'],['b']],['2'], [[['1'],['M4']],[['1'],['T6','T7']],2]],
'a04':   [[[['J1'],['18']],['1'],['a']],['1'], [[['1'],['M1','M2','M3','M4']],[['1'],['T2']],2]],
'a07':   [[[['J1'],['19']],['1'],['a']],['1'], [[['1'],['M2','M3','M4']],[['1'],['T7','T8']],2]],
'a072':   [[[['J1'],['19']],['2'],['a']],['1'], [[['1'],['M2','M3','M4']],[['1'],['T7','T8']],2]],
'a12':   [[[['J1'],['110']],['1'],['a']],['2'], [[['1'],['M1','M2','M3']],[['1'],['T2','T3','T4']],2]],
'a121':   [[[['J1'],['110']],['2'],['a']],['2'], [[['1'],['M1','M2','M3']],[['1'],['T2','T3','T4']],2]],
'a122':   [[[['J1'],['110']],['3'],['a']],['2'], [[['1'],['M1','M2','M3']],[['1'],['T2','T3','T4']],2]],
'2a12':   [[[['J1'],['110']],['1'],['b']],['2'], [[['1'],['M4']],[['1'],['T2','T3','T4']],2]],
'2a122':   [[[['J1'],['110']],['2'],['b']],['2'], [[['1'],['M4']],[['1'],['T2','T3','T4']],2]],
'a13':   [[[['J1'],['111']],['1'],['a']],['1'], [[['1'],['M2','M3','M4']],[['1'],['T9']],2]],
'a132':   [[[['J1'],['111']],['2'],['a']],['1'], [[['1'],['M2','M3','M4']],[['1'],['T9']],2]],
'a14':   [[[['J1'],['112']],['1'],['a']],['1'], [[['1'],['M2','M3','M4','M5']],[['1'],['T10']],2]],
'a142':   [[[['J1'],['112']],['2'],['a']],['1'], [[['1'],['M2','M3','M4','M5']],[['1'],['T10']],2]],
'a15':   [[[['J1'],['113']],['1'],['a']],['1'], [[['1'],['M1','M2','M3','M4']],[['1'],['T1']],2]],
'a152':   [[[['J1'],['113']],['2'],['a']],['1'], [[['1'],['M1','M2','M3','M4']],[['1'],['T1']],2]],
'a16':   [[[['J1'],['114']],['1'],['a']],['2'], [[['1'],['M2','M3']],[['1'],['T5']],2]],
'a162':   [[[['J1'],['114']],['2'],['a']],['2'], [[['1'],['M2','M3']],[['1'],['T5']],2]],
'2a16':   [[[['J1'],['114']],['1'],['b']],['2'], [[['1'],['M4']],[['1'],['T5']],2]],
'a17':   [[[['J1'],['115']],['1'],['a']],['2'], [[['1'],['M2','M3']],[['1'],['T7','T8']],2]],
'a172':   [[[['J1'],['115']],['2'],['a']],['2'], [[['1'],['M2','M3']],[['1'],['T7','T8']],2]],
'2a17':   [[[['J1'],['115']],['1'],['b']],['2'], [[['1'],['M4']],[['1'],['T7','T8']],2]],
'a08':   [[[['J1'],['116']],['1'],['a']],['2'], [[['1'],['M1','M2','M3']],[['1'],['T2','T3','T4']],2]],
'a082':   [[[['J1'],['116']],['2'],['a']],['2'], [[['1'],['M1','M2','M3']],[['1'],['T2','T3','T4']],2]],
'2a08':   [[[['J1'],['116']],['1'],['b']],['2'], [[['1'],['M4']],[['1'],['T2','T3','T4']],2]],
'a09':   [[[['J1'],['117']],['1'],['a']],['1'], [[['1'],['M2','M3','M4']],[['1'],['T9']],2]],
'a10':   [[[['J1'],['118']],['1'],['a']],['1'], [[['1'],['M2','M3','M4','M5']],[['1'],['T10']],2]],
'a19':   [[[['J1'],['119']],['1'],['a']],['1'], [[['1'],['M2','M3','M4']],[['1'],['T9']],2]],
'a20':   [[[['J1'],['120']],['1'],['a']],['1'], [[['1'],['M2','M3','M4','M5']],[['1'],['T10']],2]],
```

Figure 5-7. Transformed Operations of Part #1

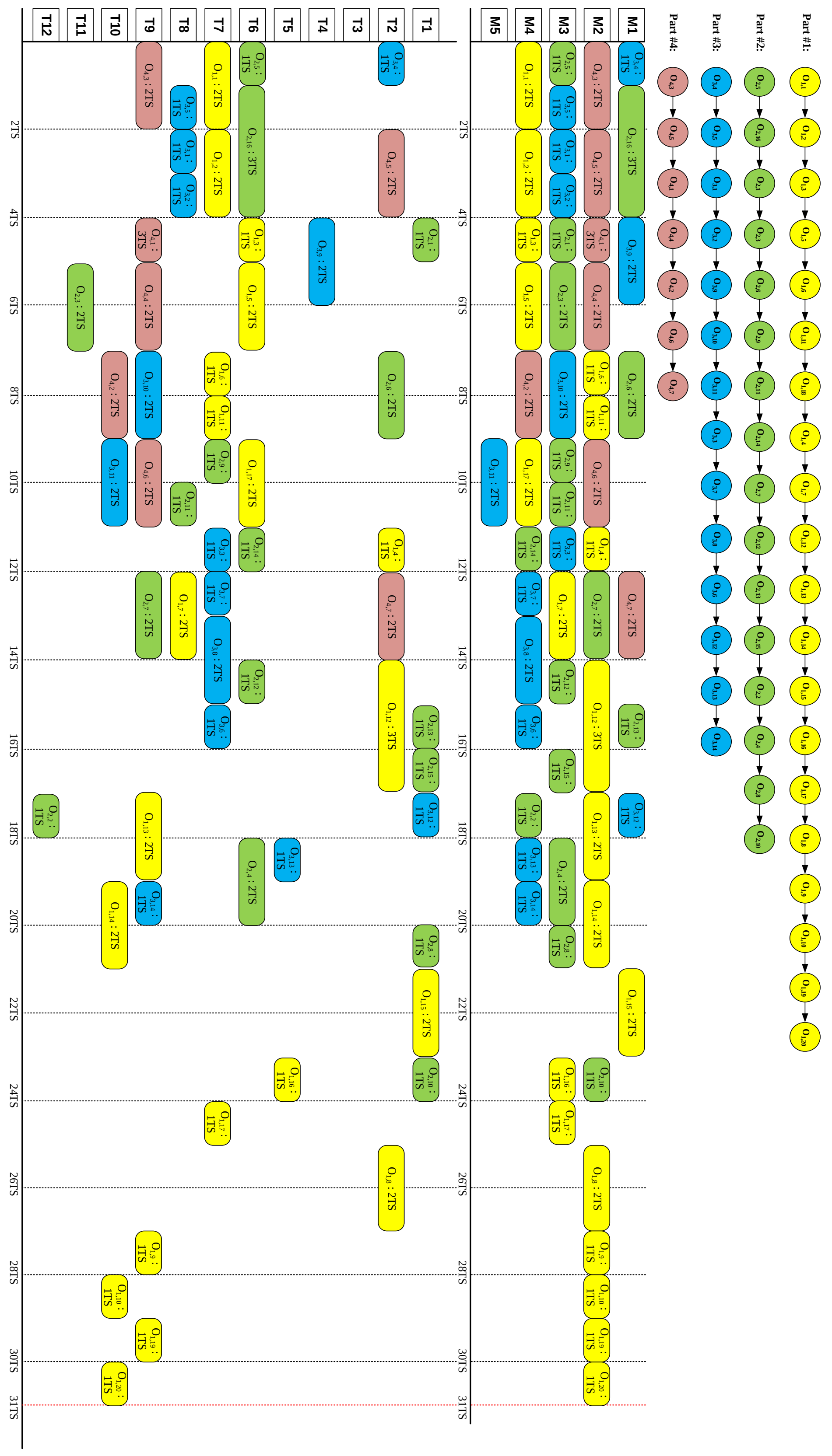


Figure 5-8. Schedule Created with Heuristics #19

Table 5-7. Outputs of Heuristics on Real-world PPS Problem

Methods	**Length Weight**	**Minimum makespan (in time slots)**	**Clock time 3-run average (s)**	**Error**	**Error rate**
Heuristics#11	LW=86	37	2533.453125	6	19.35%
Heuristics#12	LW=86	33	9576.869792	2	6.45%
Heuristics#13	LW=1	32	7474.770833	1	3.23%
Heuristics#14	LW=86	33	2542.958333	2	6.45%
Heuristics#15	LW=86	32	8680.5	1	3.23%
Heuristics#16	LW=1	32	7131.817708	1	3.23%
Heuristics#17	LW=1	33	2498.739583	2	6.45%
Heuristics#18	LW=1	33	9011.416667	2	6.45%
Heuristics#19	LW=1	31	6256.010417	0	0.00%
Heuristics#20	LW=1	53	6122.572917	22	70.97%
Heuristics#21	LW=1	36	23818.79167	5	16.13%
Heuristics#22	LW=86	32	10684.86458	1	3.23%
Heuristics#23	LW=0.001	52	4757.463542	21	67.74%
Heuristics#24	LW=1	33	21079.84375	2	6.45%
Heuristics#25	LW=86	35	9119.651042	4	12.90%
Heuristics#26	LW=1	58	4884.520833	27	87.10%
Heuristics#27	LW=0.001	35	19428.39583	4	12.90%
Heuristics#28	LW=86	32	8462.151042	1	3.23%

In Zhang et al.'s work (Zhang et al., 2014), they assume that tools are always available without causing any constraints. This assumption is based on the understanding that the machining tools are mostly available, but the machines are more critical resources in a flexible job shop. By removing the tools from the constraints, we formulate a lite version of the real-world PPS problem. The edge number is reduced to 8771, and the node number is reduced to 292 in the conflicting graph. Table 5-8 shows the outputs of Heuristics #11 to Heuristics #28 on this simplified real-world PPS problem. Note that the optimum solution for this instance is also 31 time slots; it is calculated by manipulating the IP model in a trial and error fashion. The results with an error rate of less than 5% take less than 700 seconds clock time for finishing the computation. Although our approach almost doubles the computation time compare to Zhang et al.'s work, the runtime is still acceptable. We can say that our approach has acceptable practicability and feasibility on real-world PPS problem. To further justify this conclusion, the following section discusses the details regarding the scalability and accuracy of the proposed approach.

Table 5-8. Outputs of Heuristics on Real-world PPS Problem without Tool Constraints

Methods	**Length Weight**	**Minimum makespan (in time slots)**	**Clock time 3-run average (s)**	**Error**	**Error rate**
Heuristics#11	LW=86	37	147.2188	6	19.35%
Heuristics#12	LW=1	35	704.8594	4	12.90%
Heuristics#13	LW=86	33	521.4063	2	6.45%
Heuristics#14	LW=1	33	134.8698	2	6.45%
Heuristics#15	LW=1	32	671.2188	1	3.23%
Heuristics#16	LW=1	35	467.6979	4	12.90%
Heuristics#17	LW=1	34	124.8281	3	9.68%
Heuristics#18	LW=1	32	686.7813	1	3.23%
Heuristics#19	LW=1	34	449.1823	3	9.68%
Heuristics#20	LW=1	43	287.8906	12	38.71%
Heuristics#21	LW=0.001	35	1629.25	4	12.90%
Heuristics#22	LW=86	32	580.8958	1	3.23%
Heuristics#23	LW=1	43	293.9688	12	38.71%
Heuristics#24	LW=0.001	35	1671.74	4	12.90%
Heuristics#25	LW=1	31	591.6198	0	0.00%
Heuristics#26	LW=1	43	283.3698	12	38.71%
Heuristics#27	LW=1	36	1806.609	5	16.13%
Heuristics#28	LW=1	33	598.1563	2	6.45%

5.3 Results and Discussions on Test Instances

We create nineteen test instances based on the structure of the real-world PPS example problem with randomized sequencing constraints and resource combinations. The detailed input information is in Appendix III. We run each heuristics configuration on each test instance for three times, and details of the results are shown in Appendix IV and Appendix V. Since our approach returns feasible results on all the test instances and the real-world example, we assume that our approach has a satisfactory robustness on similar types of the PPS problem. Then, the discussion is focusing on the scalability and accuracy. The scalability analysis shows how the proposed approach behaves on different size and variance of the inputs. It can be evaluated based on the computation time versus the different input sizes, node numbers, and edge numbers of the different conflicting graphs. The accuracy refers to how likely the proposed approach can get to the optimum solution. It can be measured by the average and maximum error rate of all the test instances.

5.3.1 Scalability

The essential understanding of our approach to PPS problems, MWIS algorithms are the determinant of the computation speed of different heuristics configurations. For those heuristics configurations based on the same MWIS algorithm, the ones with more complex weight factor calculations are slower. But this difference is minimal.

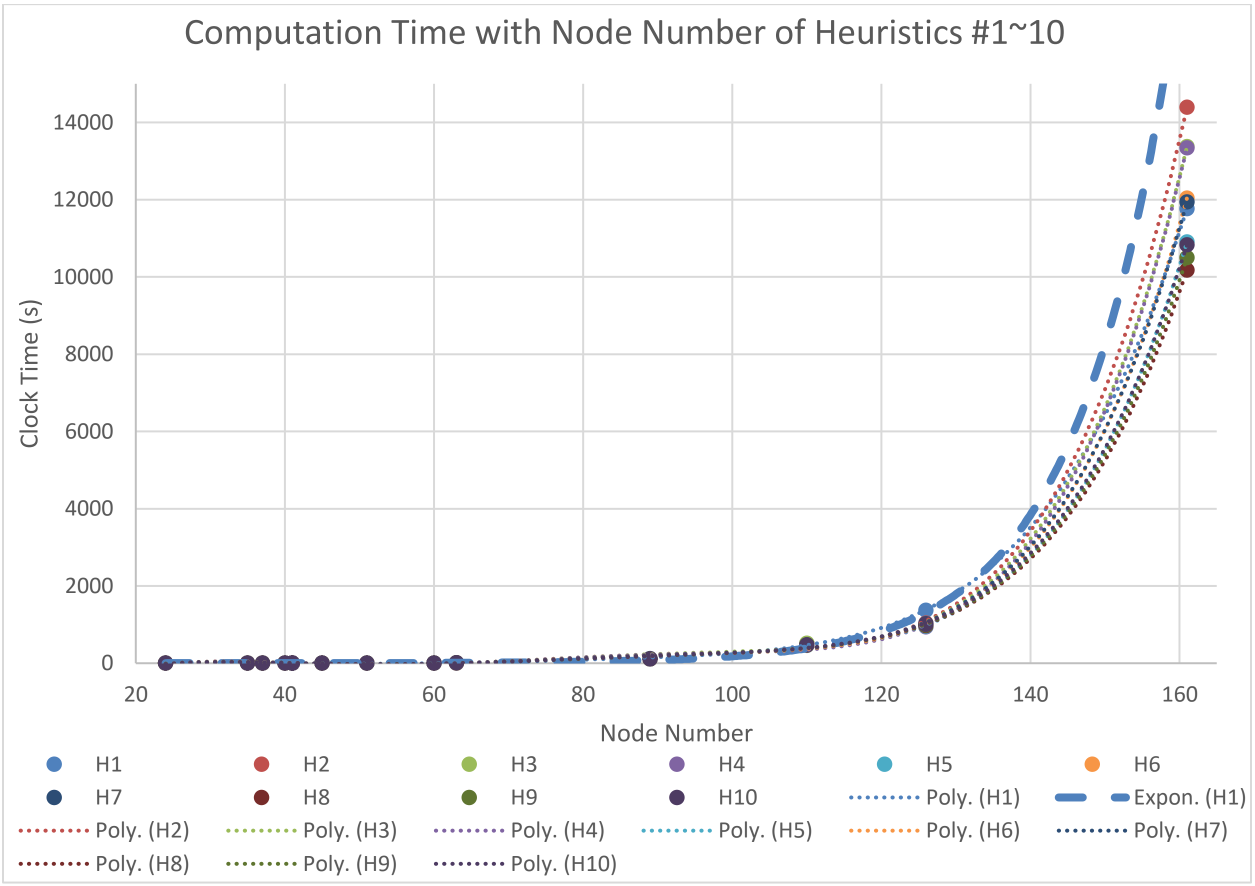


Figure 5-9. Computation Time with Node Number of Heuristics #1~10

Figure 5-9 and Figure 5-10 show how the computation time is changing with node number and edge number on Heuristics #1~10, respectively. The Heuristics #1~10, which are based on the two exact MWIS algorithms, Algorithm A1 MWIS and Algorithm A2 AMISL, are much slower than all other heuristics configurations. The computation time could be hours when there are about 140 nodes and 4000 edges, which could be much smaller than a typical PPS problem. Although the worst case of the two algorithms can be exponentially slow, the PPS problem considered here may not always be the worst case. As shown in Figure 5-9 and Figure 5-10, the Heuristics #1~10 match higher-order (order 4 or higher) polynomial trendlines, but they are faster than the exponential trendline.

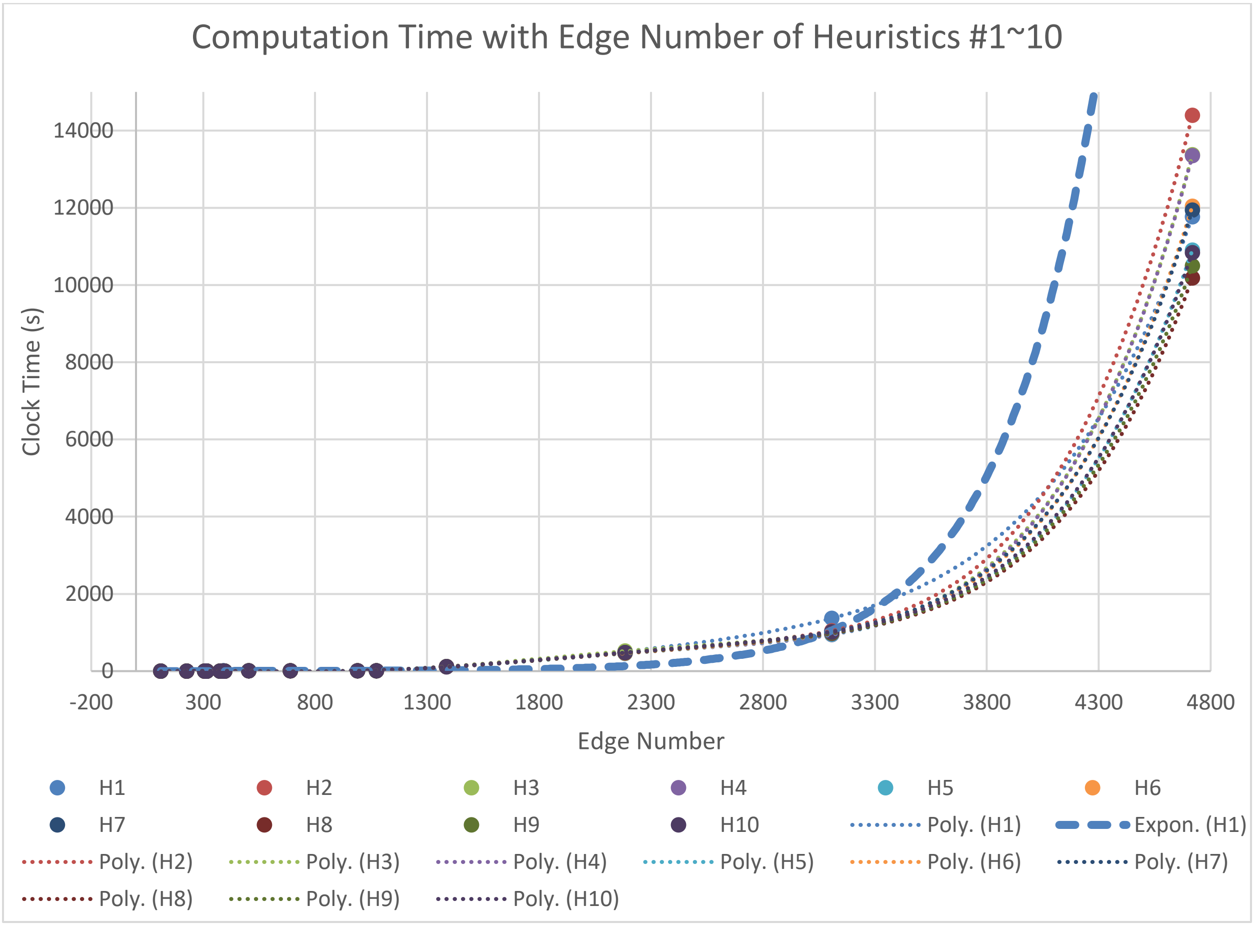


Figure 5-10. Computation Time with Edge Number of Heuristics #1~10

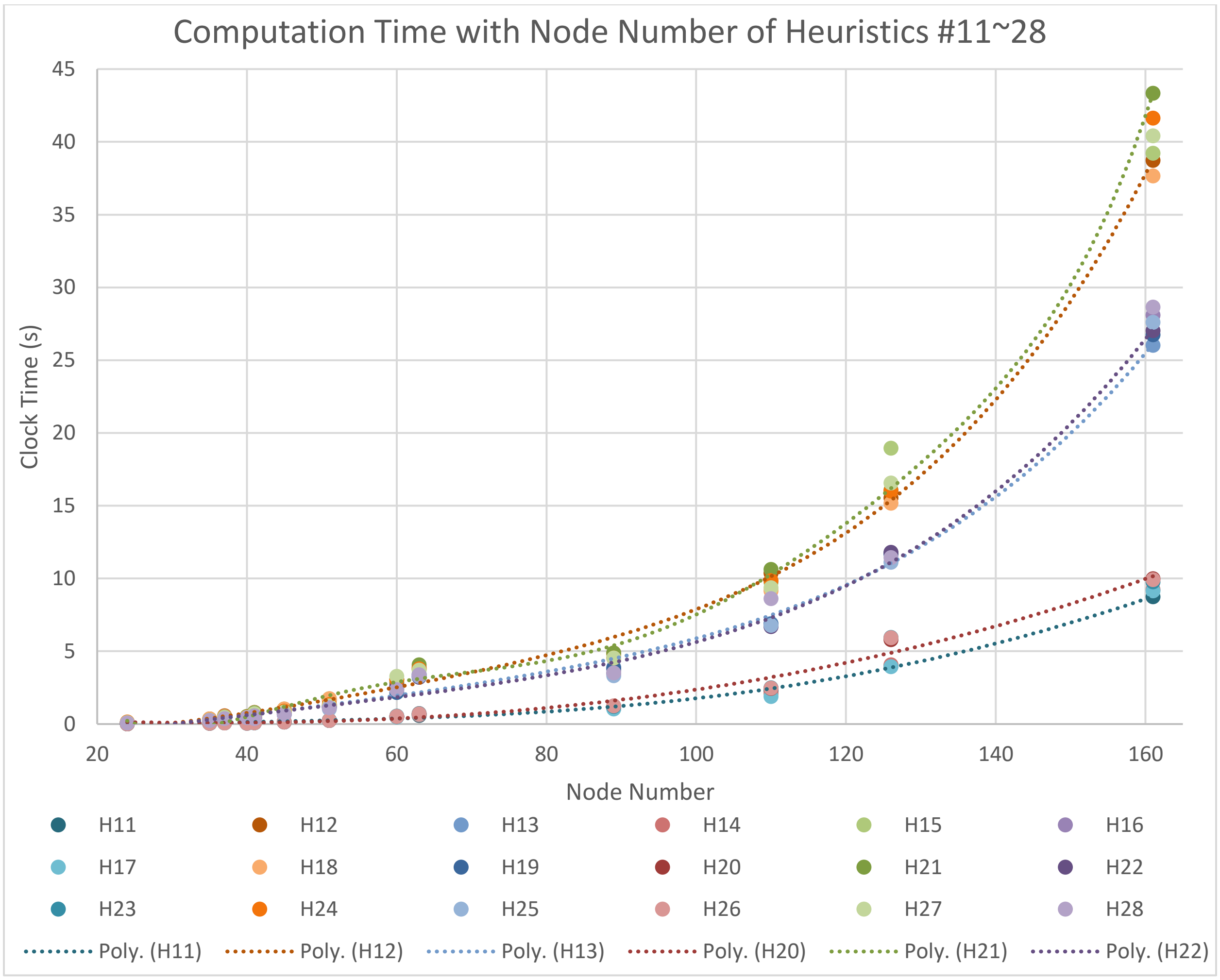


Figure 5-11. Computation Time with Node Number of Heuristics #11~28

For Heuristics #11~28, how the computation time is changing with node number and edge number is represented in Figure 5-11 and Figure 5-12, respectively. The Heuristics #11~28 are based on the approximation MWIS algorithms, GWMIN, GWMIN2, and their combinations. Heuristics #11, Heuristics #14, Heuristics #17, Heuristics #20, Heuristics #23, and Heuristics #26, which utilizing the Algorithm A3 GWMIN and Algorithm A6 GWMIN2 are the fastest. Heuristics #13, Heuristics #16, Heuristics #19, Heuristics #22, Heuristics #25 and Heuristics #28, which utilizing Algorithm A5 MWIS_SubCS_GWMIN and Algorithm A8 MWIS_SubCS_GWMIN2 are following. Heuristics #12, Heuristics #15, Heuristics #18, Heuristics #21, Heuristics #24 and Heuristics #27, which utilizing Algorithm A4 MWIS_CS_GWMIN and Algorithm A7 MWIS_CS_GWMIN2 are the slowest. The computational speed of these Heuristics follows the similar trendlines of the approximation MWIS algorithms, as discussed in Chapter 3. And Heuristics based on approximation MWIS algorithms are much feasible in the sense of computation time.

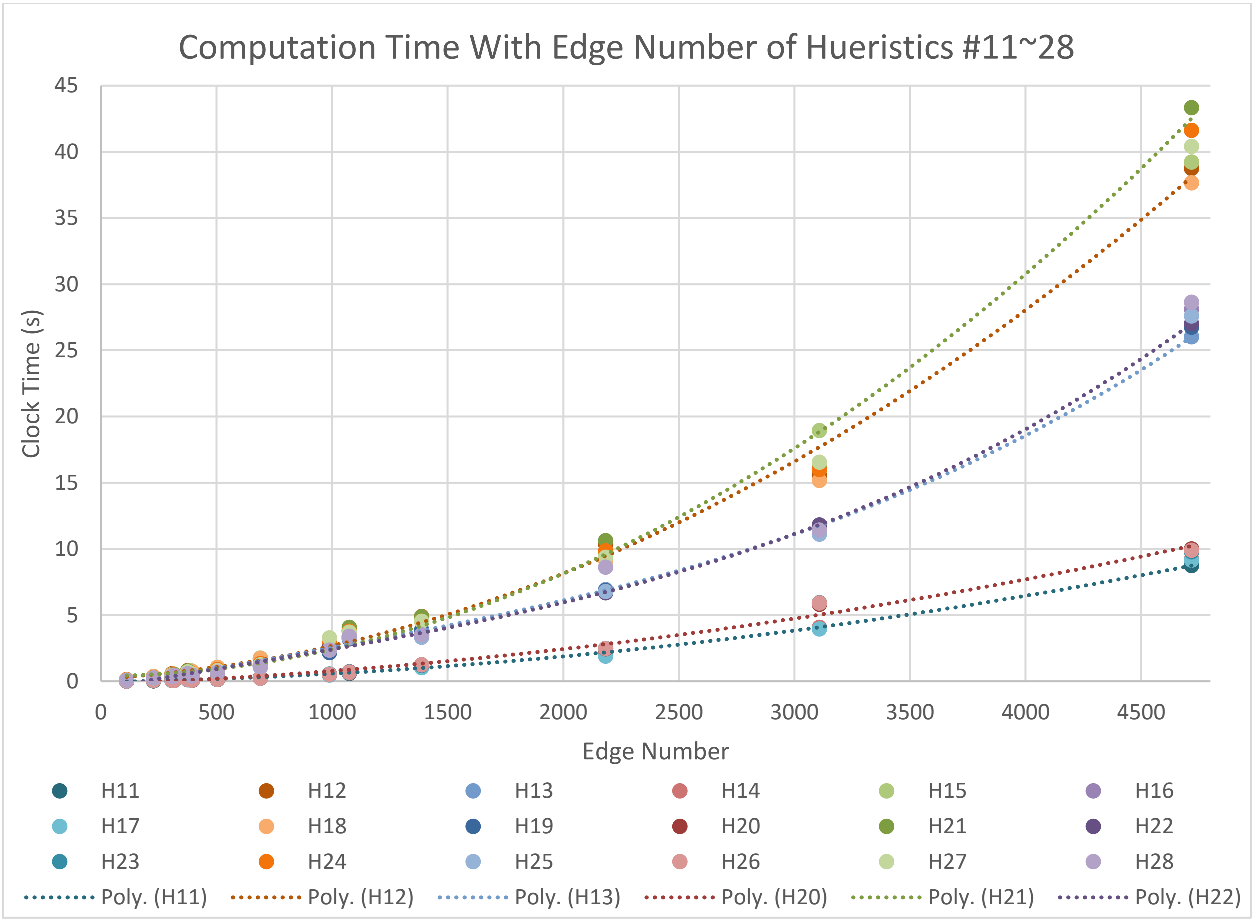


Figure 5-12. Computation Time with Edge Number of Heuristics #11~28

5.3.2 Accuracy

Figure 5-13 shows the average and maximum error rate for all heuristics configurations. The detailed information of the Heuristics configurations is as Table 4-3. The detailed accuracy summary of all the heuristics configurations on all tests is in Appendix IV. The detailed input information and the detailed results of each test instance is in Appendix III and Appendix V, respectively.

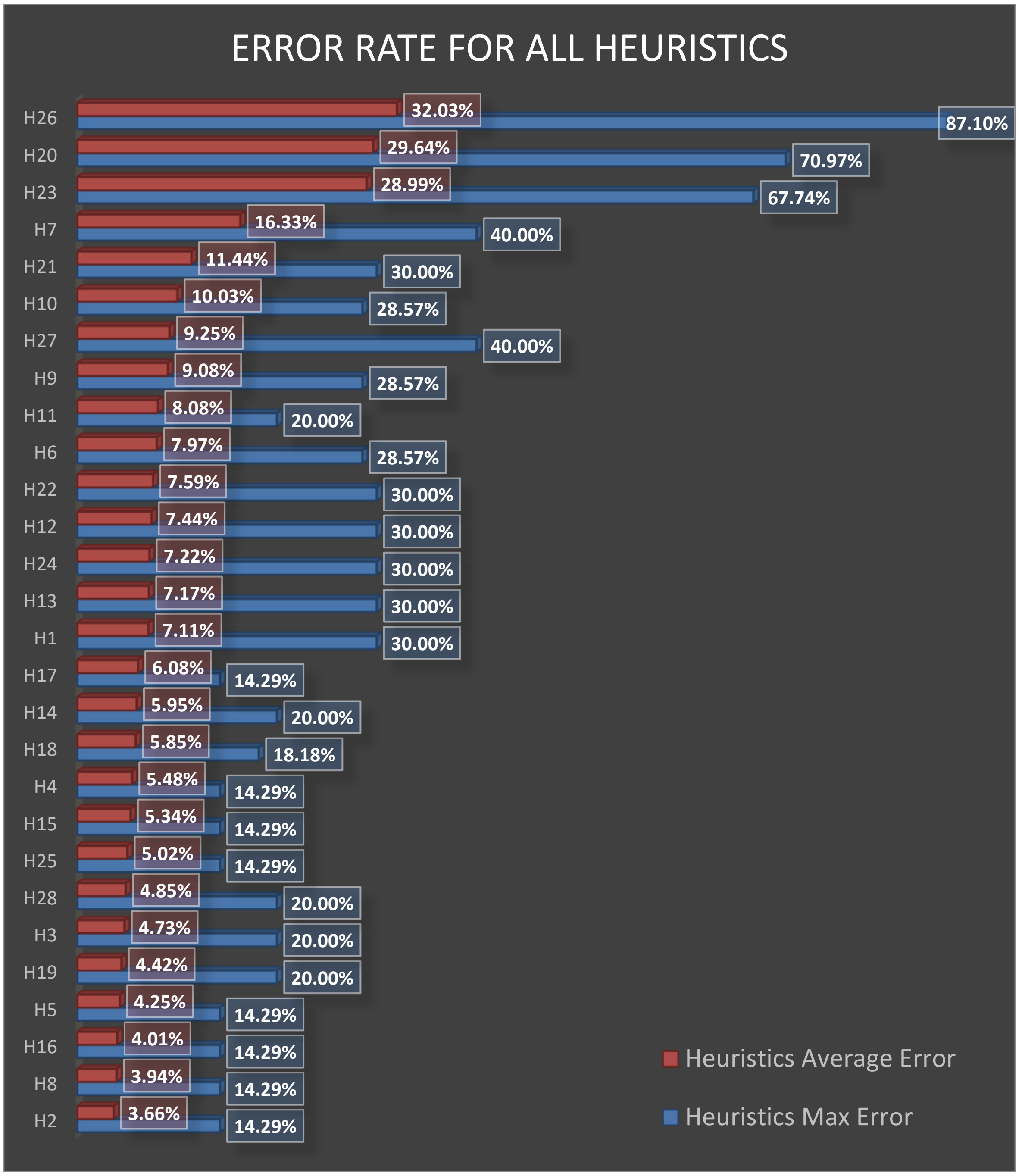


Figure 5-13. The Average and Maximum Error Rate for All Heuristics Configurations

Assume $TS_{optimum}$ is the minimum number of time slots need for the PPS problem on the test instance, and TS is the number of time slots found by our approach. The error rate is calculated using the function below.

$$Error\ Rate = \frac{TS - TS_{optimum}}{TS_{optimum}} \times 100\%$$

Note that the $TS_{optimum}$ is calculated based on the IP model with manipulating inputs to get the optimum result with reasonable computation time, and the error rate of each heuristics configuration is calculated based on the best accuracy among the three different length weight factors.

Let the threshold for heuristics configuration selection be the average error of less than 7% and the maximum error of less than 20%. For Heuristics #1-10 with the exact MWIS algorithms, from the best to the worst, Heuristics #2, Heuristics #8, Heuristics #5, Heuristics #3 and Heuristics #4 are satisfactory. For Heuristics #11-28 with the approximation MWIS algorithms, from the best to the worst, Heuristics #16, Heuristics #19, Heuristics #28, Heuristics #25, Heuristics #15, Heuristics #18, Heuristics #14, and Heuristics #17 are satisfactory.

Based on the computational experiments in Chapter 3, the general accuracy of the MWIS algorithms can be listed below from the best accuracy to the worst:

- Algorithm A1 MWIS
- Algorithm A2 AMISL (same as Algorithm MWIS)
- Algorithm A5 MWIS_SubCS_GWMIN
- Algorithm A8 MWIS_SubCS_GWMIN2
- Algorithm A4 MWIS_CS_GWMIN
- Algorithm A3 GWMIN

- Algorithm A7 MWIS_CS_GWMIN2
- Algorithm A6 GWMIN2

Compare with the results shown in Figure 5-13, with the same weight factors assignment, a more accurate MWIS algorithm leads to a better accuracy output of the PPS problem. None of the satisfactory heuristics is using the least accurate MWIS algorithms, Algorithm A7 MWIS_CS_GWMIN2 and A6 GWMIN2. In other words, while using the proposed approach for the PPS problem, a relatively accurate MWIS algorithm is required. This is the evidence of the necessity of the better accuracy MWIS algorithms proposed in Chapter 3.

$$\text{Job \#1: } \frac{T_{11}[(R_1,R_2,R_3)_2]}{2} \rightarrow \frac{T_{12}[(R_1,R_2,R_3)_2]}{1} \rightarrow \frac{T_{13}[(R_1,R_2,R_3)_1]}{2} \rightarrow \frac{T_{14}[(R_4)_1]}{3}$$

$$\text{Job \#2: } \frac{T_{21}[(R_1,R_2,R_3)_1]}{1} \rightarrow \frac{T_{22}[(R_1)_1\text{and}(R_2,R_3)_1]}{2} \rightarrow \frac{T_{23}[(R_4)_1]}{3}$$

$$\text{Job \#3: } \frac{T_{31}[(R_1,R_2)_1]}{1} \rightarrow \begin{pmatrix} \frac{T_{32a}[(R_1,R_2)_2]}{1} \\ \frac{T_{32b}[(R_3)_1]}{2} \end{pmatrix}^{1} \rightarrow \frac{T_{33}[(R_3,R_4)_1]}{3}$$

$$\text{Job \#4: } \frac{T_{41}[(R_1,R_2,R_3)_2]}{2}$$

$$\text{Job \#5: } \frac{T_{51}[(R_1,R_2,R_3)_2]}{2}$$

Figure 5-14. Details of Test Instances T24

The above-mentioned heuristics configurations may not able to reach the optimum results on some of the test instances. These bad instances are T6, T11, T12, T13, T14, T17, T18, T19, T20, and T24. Figure 5-15 shows the average and maximum error rate for all heuristics configurations on these bad instances. These instances have concentrated resource requirements. Let us take the instance T24 (Figure 5-14) as an example. The jobs in the instances have a significant difference in the number of time slots for finishing. Also, the beginning Unit Tasks are concentrated on the resources R_1, R_2, and R_3, and the ending Unit Tasks are concentrated on the resources R_4. Since the MWIS algorithm tries to schedule as many nodes as possible, it may cause the ending Unit Tasks all leftover, but they cannot be processed on parallel machines. We iterate the three levels of length weight coefficient, median, high and low, as LW_c^M, LW_c^H and LW_c^L, respectively with the proposed heuristics configurations to balance the length of each job and the concentrated resources requirements. So that the maximum error rate of each satisfactory heuristics configuration is not exceeding 20%.

Another interesting finding is that the Heuristics #14 and #17, which are using the approximation algorithms GWMIN, perform well on these bad test instances. The hypothesis is that the GWMIN generates the selection of the node with the maximum weight. This may avoid the concentrating resources blocking the optimum results.

Based on the discussions on scalability and accuracy, the better heuristics configurations for the PPS problem are listed as below,

- Heuristics #16, Algorithm MWIS_SubCS_GWMIN, MWIS A2: MWIS Weights 2
- Heuristics #19, Algorithm MWIS_SubCS_GWMIN, MWIS A3: MWIS Weights 3
- Heuristics #28, Algorithm MWIS_SubCS_GWMIN2, MWIS A3: MWIS Weights 3
- Heuristics #25, Algorithm MWIS_SubCS_GWMIN2, MWIS A2: MWIS Weights 2
- Heuristics #15, Algorithm MWIS_CS_GWMIN, MWIS A2: MWIS Weights 2
- Heuristics #18, Algorithm MWIS_CS_GWMIN, MWIS A3: MWIS Weights 3
- Heuristics #14, Algorithm GWMIN, MWIS A2: MWIS Weights 2
- Heuristics #17, Algorithm GWMIN, MWIS A3: MWIS Weights 3

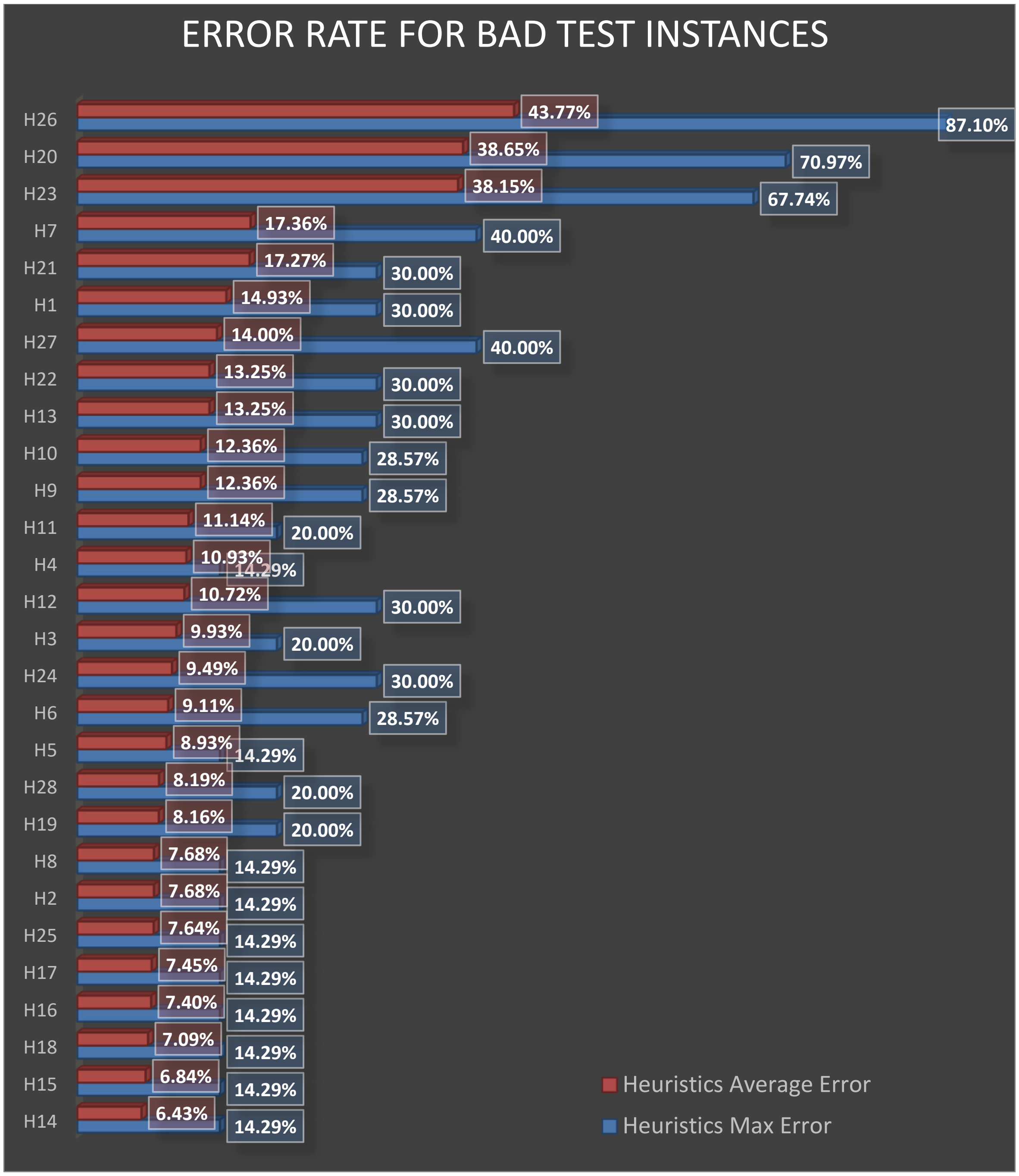


Figure 5-15. The Average and Maximum Error Rate on Bad Test Instances

5.4 Summary

In this chapter, we verify the practicability and feasibility of the proposed approach for the PPS problem on a real-world example and further test instances. The implementation of our approach is error-free and bug-free on all the tests. The IP model described in Chapter 4 is implemented and tested. Although it is not feasible for solving the PPS problem at a realistic computational speed, it can be used to verify the optimum solution with conditions.

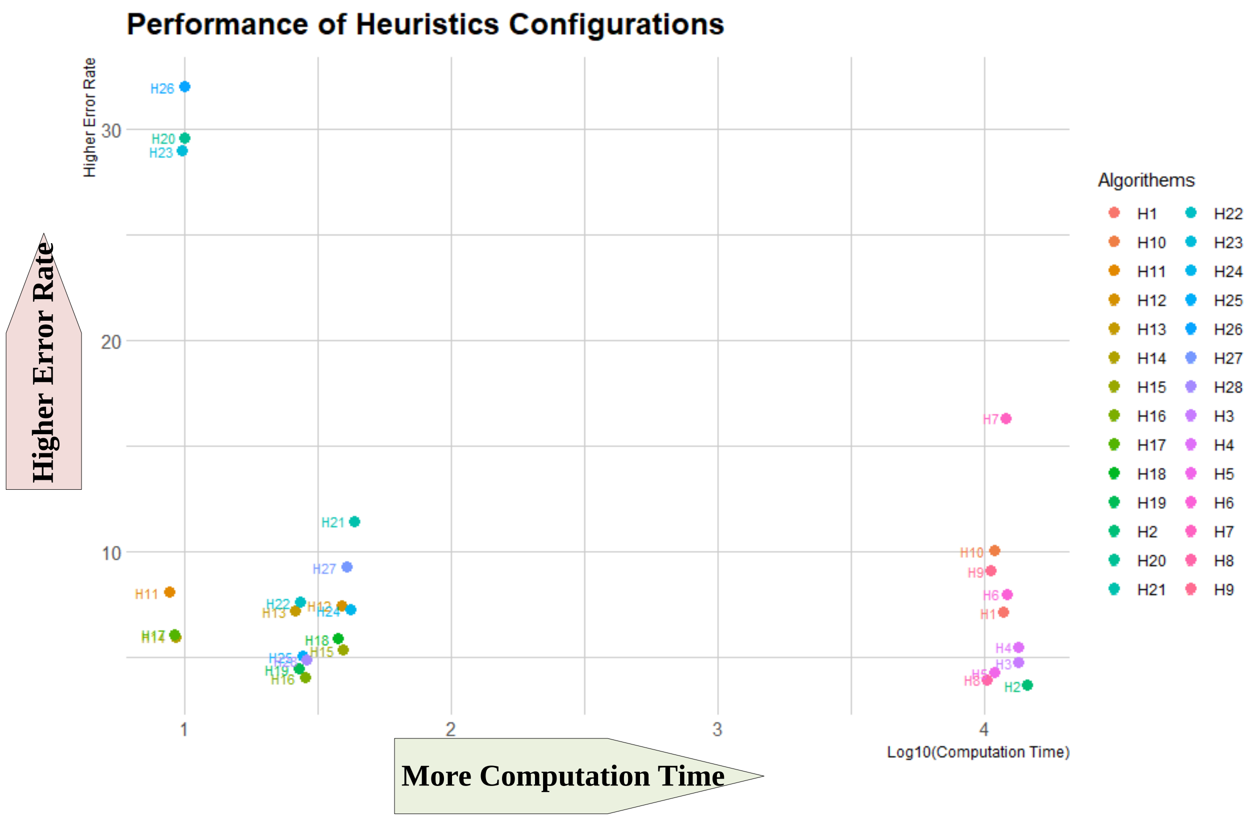


Figure 5-16. Performance of the Heuristics Configurations

The test results of all heuristics configurations on all test instances are reported and analyzed in terms of the scalability and accuracy. Figure 5-16 is the summary of the performance of the heuristics configurations. The test results also justify the statement that better accuracy and faster MWIS algorithms are desired for solving the PPS problem when using our approach. The satisfactory heuristics configurations for general cases are Heuristics #16, Heuristics #19, Heuristics #28, Heuristics #25, Heuristics #15, Heuristics #18, Heuristics #14, and Heuristics

#17 in an accuracy order. For the cases with limited size, some of the heuristics configurations using exact MWIS algorithms can also be considered. These heuristics configurations are Heuristics #2, Heuristics #8, Heuristics #5, Heuristics #3 and Heuristics #4. All these heuristics configurations considered as satisfactory have the average error of less than 7% and the maximum error of less than 20%.

Chapter 6. Conclusions

This chapter concludes the dissertation and discusses the contributions and future work of this research. In particular, the two main contributions, (1) algorithms for the **M**aximum **W**eighted **I**ndependent **S**et (MWIS) problem and (2) a novel approach for the **P**rocess **P**lanning and **S**cheduling (PPS) problem, are described in section 6.1 and section 6.2, respectively. The main research contributions are highlighted in section 6.3. Lastly, possible future directions for improving and extending the work presented in this dissertation are discussed in section 6.4.

6.1 Algorithms for Maximum Weighted Independent Set (MWIS) Problem

6.1.1 Development of MWIS Algorithms

This research considers the MWIS problem on general graphs and develops algorithms for solving the MWIS problem in a divide and conquer structure. In order to reduce the complexity of the algorithm structure to the greatest extent, utility functions are developed or adopted; they are Algorithm 3-1: the basic cycles algorithm (Paton, 1969), Algorithm 3-2, the diameter algorithm (Takes & Kosters, 2011, 2013; Borassi et al., 2015), and Algorithm 3-3: the middle node algorithm.

Based on the divide and conquer structure, two exact MWIS algorithms, Algorithm A1 MWIS and Algorithm A2 AMISL, are developed. Moreover, faster approximation algorithms can be composed based on Algorithm A1. In this case, the complexity of the proposed algorithm can be reduced, and the accuracy of approximation algorithms can be improved. We implement two approximation algorithms from literature, Algorithm A3 GWMIN and Algorithm A6 GWMIN2

(Sakai et al., 2003), and developed the following algorithms by merging them with Algorithm A1 to improve their accuracy.

- Algorithm A4 MWIS_CS_GWMIN: it is an algorithm composed of Algorithm A1 and Algorithm A3. This algorithm computes Compare Sets based on the whole induced subgraph at each level using Algorithm A3 GWMIN.
- Algorithm A5 MWIS_SubCS_GWMIN: it is an algorithm composed of Algorithm A1 and Algorithm A3. This algorithm computes Compare Sets based on the induced CSSs, excluding the current removed node, using Algorithm A3 GWMIN.
- Algorithm A7 MWIS_CS_GWMIN2: it is an algorithm composed of Algorithm A1 and Algorithm A6. This algorithm computes Compare Sets based on the whole induced subgraph at each level using Algorithm A6 GWMIN2.
- Algorithm A8 MWIS_SubCS_GWMIN2: it is an algorithm composed of Algorithm A1 and Algorithm A6. This algorithm computes Compare Sets based on the induced CSSs, excluding the current removed node, using Algorithm A6 GWMIN2.

6.1.2 Performance of MWIS Algorithms

All eight algorithms are tested on the test instances, which are based on the PPS application environment. The details of the test results are shown in Appendix II.

Algorithms A1 and A2 are the two exact algorithms for computing the MWIS of a weighted undirected graph. They are of high complexity. The computation time can be hours when there are about 140 nodes and 4000 edges in the conflicting graph. Although the worst case of Algorithms A1 and A2 can be exponentially slow, the application scenarios of the PPS problem considered in this research may not always be the worst case. In the test instances, Algorithms

A1 and A2 match higher-order (order 4 or higher) polynomial trendlines, and they are actually faster than the exponential trendline.

Algorithms A3 and A6 are low-complexity greedy algorithms for the MWIS problem from literature (Sakai et al., 2003). They are the fastest among the 8 algorithms discussed in this research. The computation time is less than half-second on all test instances. Algorithms A3 has a nearly linear or log-linear complexity on the test instances, and Algorithms A6 has a polynomial complexity. This difference in complexity is due to the different node-selecting functions of the two algorithms.

Algorithms A4, A5, A7, and A8 are the composed approximation algorithms based on Algorithms A1. They are slower than the approximation algorithms from the literature, but still much faster compared to the two exact MWIS algorithms. The computation time is less than 45 seconds on all the test instances. In general, a faster approximation algorithm leads to a faster composed algorithm; and while composing the algorithms, applying the approximation algorithm on smaller subgraphs leads to a faster composed algorithm. In terms of the accuracy of the MWIS algorithms, composing the approximation algorithms with Algorithms A1 can improve the accuracy. While composing, applying the approximation algorithm on smaller subgraphs, for our case, the induced CSSs, can achieve better accuracy. The general accuracy of the best five algorithms can be listed below from the best to the worst:

- Algorithm A1 MWIS
- Algorithm A2 AMISL (same as Algorithm MWIS)
- Algorithm A5 MWIS_SubCS_GWMIN
- Algorithm A8 MWIS_SubCS_GWMIN2
- Algorithm A4 MWIS_CS_GWMIN

Note that all these algorithms considered satisfactory have the average error of less than 1% and the maximum error of less than 13% (The first four algorithms have the maximum error less than 9%) on all test instances.

6.2 Approach for Process Planning and Scheduling (PPS) Problem

This dissertation considers a general type of PPS problem, and proposes a novel approach for formulating and solving the resource-constrained PPS optimization problem. In our approach, the two procedures, the resource selection and process scheduling, of the PPS problem are integrated. The PPS problem is formulated into an undirected weighted conflicting graph due to the nature of sequencing and resource constraints. A node in the conflicting graph represents one operation with one possible combination of its required resources during one time slot, and an edge indicates that there is a conflict between the two nodes at both ends of the edge. Each node in the graph is assigned with a weight factor as the guidance for the operation and resource selections to fulfill the optimization objective. The nodes with a higher possibility leading to the objective are given priority when generating the schedule. The schedule with resource allocations is generated by solving the MWIS problem of the graph.

Twenty-eight heuristics configurations for solving the PPS problem are generated by combining the seven weight factor arrangements with the eight MWIS algorithms. With careful consideration on scalability and accuracy, the best heuristics configurations for the PPS problem are listed as below,

- Heuristics #16, Algorithm MWIS_SubCS_GWMIN, MWIS A2: MWIS Weights 2
- Heuristics #19, Algorithm MWIS_SubCS_GWMIN, MWIS A3: MWIS Weights 3
- Heuristics #28, Algorithm MWIS_SubCS_GWMIN2, MWIS A3: MWIS Weights 3

- Heuristics #25, Algorithm MWIS_SubCS_GWMIN2, MWIS A2: MWIS Weights 2
- Heuristics #15, Algorithm MWIS_CS_GWMIN, MWIS A2: MWIS Weights 2
- Heuristics #18, Algorithm MWIS_CS_GWMIN, MWIS A3: MWIS Weights 3
- Heuristics #14, Algorithm GWMIN, MWIS A2: MWIS Weights 2
- Heuristics #17, Algorithm GWMIN, MWIS A3: MWIS Weights 3

Note that all these heuristics configurations, which are considered satisfactory, have the average error of less than 7% and the maximum error of less than 20% on the test cases.

6.3 Research Contribution

The first main contribution is on the MWIS problem. This work proposes a divide and conquer algorithm structure with relatively low time complexity for solving the MWIS problem. The exact MWIS algorithm and **A**ll **M**aximal **I**ndependent **S**et **L**isting (AMISL) algorithm are developed based on this algorithm structure. The proposed algorithm structure can also be used to compose the exact MWIS algorithm with existing approximation MWIS algorithms for compromises on accuracy and computational speed. Utilizing existing approximation algorithms with the proposed algorithm structure is an effective way to improve the accuracy of existing approximation MWIS algorithms. All eight algorithms for the MWIS problem, the exact MWIS algorithm, the AMISL algorithm, two approximation algorithms from the literature, and four composed algorithms, are tested on the test instances based on the PPS application environment. A set of "good-performance" MWIS algorithms are highlighted based on the test results.

The second main contribution is on the PPS problem. Unlike the commonly used iteration type of approaches, such as generic algorithms and metaheuristics, or the mixed-integer programming approaches, our approach provides a different angle to address the PPS problem and shows advantages over other approaches as illustrated in Table 6-1. The PPS problem is formulated as a conflicting weighted graph and generating the integrated process schedule with resource allocation by solving the MWIS problem. This idea extends the universality of the formulation of the graph coloring based scheduling. The new approach requires minimum iteration. And it is guaranteed to return a feasible solution due to the nature of solving the MWIS problem on a conflicting weighted graph. The new approach computes the schedule of each time slot separately. We develop different weight factor calculation strategies and arrangements as the guidance for achieving the optimization objective. With carefully defined weight factors and "good-performance" MWIS algorithms, the new approach has satisfactory accuracy and computational speed. A set of "good-performance" heuristics configurations are found based on the test results.

Table 6-1. Comparing the New Approach with Other Methods*

Measurements	**Generic Algorithms**	**Simulated Anneal**	**Tabu Search**	**Mixed-integer Programming**	**Partial Solutions**	**Graph Coloring Scheduling**
Accuracy	=	=	=	-	+	NA
Computational speed	=	=	=	+	=	NA
Universality	-	-	=	-	NA	+
Dependence on iterations	+	+	+	+	+	NA
Feasibility	+	+	+	=	+	NA
Separated solutions of each time slot	+	+	+	+	+	NA

*'+': The new approach is better on the measurement compare with the other method.
'=': The new approach is similar or potentially better compare with the other method.
'-': The new approach is not as good as the other method.
'NA': It is hard to compare the new approach with the other method.

6.4 Future Work

In this research, we attempt to address the two classic problems, the MWIS problem and the PPS problem in universality. As the review shown in Chapter 2, we can have broad applications by solving the two problems. In this section, we are focusing on (1) the potential improvements and extensions of current work in the scheduling domain; and (2) integrating the solution to the PPS problem with Smart Manufacturing infrastructure, the **S**mart **P**roduct **L**ifecycle **M**anagement (sPLM) system.

6.4.1 Improvements and Extensions

To speed up the computation:

For the MWIS algorithms, the divide and conquer algorithm structure can be transformed into multi-threading. Each connected subgraph after node removal are independent. The computation of the connected subgraphs can be assigned to different threads.

For the formulation of the PPS problem, Unit Tasks of the operations that are constrained to be processed in the far future (a good number of time slots later) may have a very limited impact on the scheduling of earlier time slots. While generating the conflicting graph, we may only consider the most recent several Unit Tasks of each part so that the size of the conflicting graph can be reduced.

To improve the accuracy:

The current weight calculation and weight factor arrangements can be fine-tuned and closely-integrated with the MWIS algorithms based on the part and resource information to achieve better node selections. The examples can be specific heuristics for weight calculation, machine learning methods to optimize the value of the weight factor.

Stochastic **O**ptimization (SO) methods are optimization methods that generate and use random variables (Spall, 2003). This method can be applied to bring in probabilistic in the schedule generation process when solving the MWIS problem for each time slot. It enables the possibility of iteratively selecting different sets of nodes for each time slot. By applying this method, the trapping of bad node selections may be avoided.

To improve the universality:

Our approach for the PPS problem can be easily implemented for a dynamic job taking environment by updating the conflicting graph for each time slot. The traditional approach requires taking consideration of known operations and iterates to get an optimum schedule for recent periods, which requires searching in a vast solution space. Unlike iteration-based approaches, the new approach computes the schedule of each time slot separately, which may only require partial operation information of each job. And for each time slot, the new approach tries to utilize the resources as much as possible by solving the MWIS problem.

Our approach for the PPS problem can be easily implemented with the flexible operation sequencing constraints by updating the conflicting graph for each time slot. In this case, all the Unit Tasks that are not restricted by the sequencing constraints are considered as Unit Task Candidates to be selected by solving the MWIS problem.

The conflicting weighted graph may be extended to a multi-connected graph, directed graph, weighted edges to represent more information for the optimization problem modeling. And further, we wish to improve the approach by, such as enabling the multi-objective optimization, introducing more variables for the details of the PPS problem, introducing probabilistic variables, and more.

6.4.2 Integration with the sPLM System

The sPLM system is developed in the Knowledge Engineering Laboratory at Syracuse University. It is a platform developed based on an open-source **P**roduct **L**ifecycle **M**anagement (PLM) system, Aras Innovator, to handle product lifecycle information to support decisionmaking processes (Li, 2018). Figure 6-1 shows that a lot of resource-constraint scheduling problems naturally arise in the application of the sPLM system as its prescriptive analytics capability (Sun et al., 2017). Prescriptive analytics, referred to as the "final frontier of analytic capabilities (Gartner, 2017)," it entails the application of mathematical and computational sciences and suggests decision options to take advantage of the results of descriptive and predictive analytics (Basu, 2013; Engel et al., 2012; Lepenioti, 2020). The scenario is that knowledge such as resource management, materials management, and product development from the sPLM system can be integrated and formulate into solution nodes and constraint edges for detailed and adaptive planning and scheduling. Such a conflicting graph can be used for solving different scheduling problems, like delivery planning, production planning, product development planning, and more.

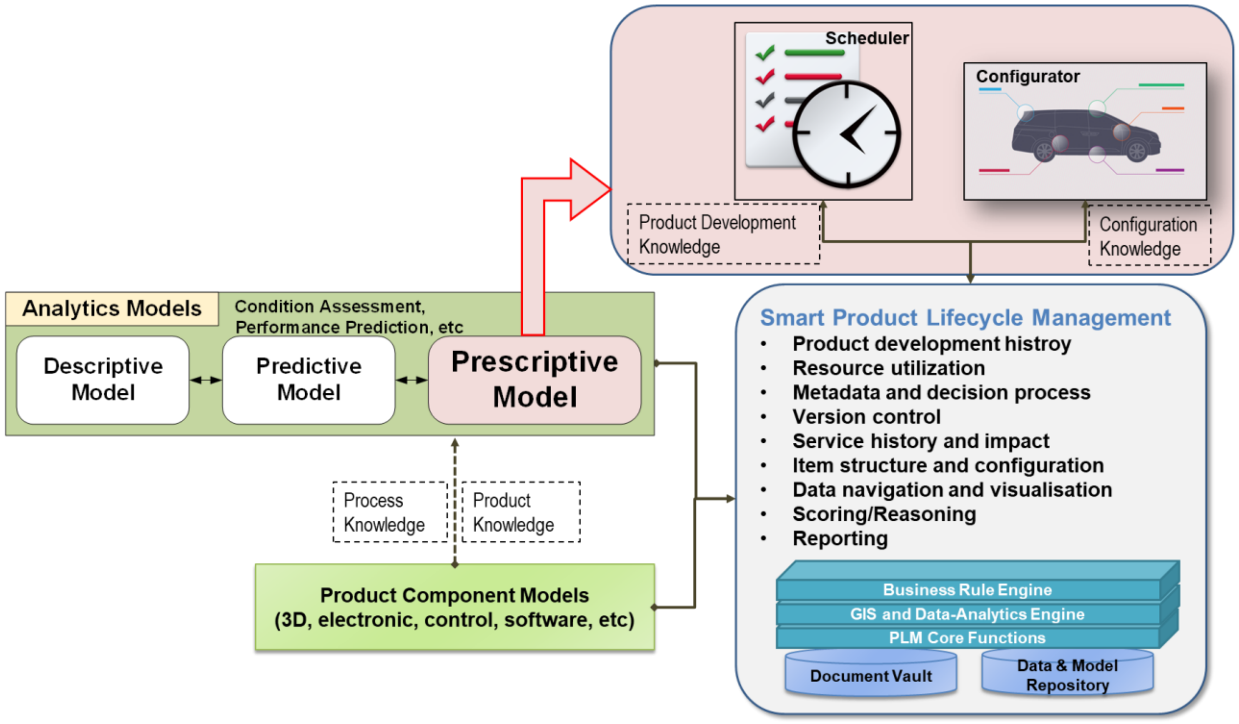

Scheduler
Configurator
Product Development Knowledge
Configuration Knowledge
Analytics Models
Condition Assessment, Performance Prediction, etc
Descriptive Model
Predictive Model
Prescriptive Model
Process Knowledge
Product Knowledge
Product Component Models (3D, electronic, control, software, etc)
Smart Product Lifecycle Management
Product development histroy
Resource utilization
Metadata and decision process
Version control
Service history and impact
Item structure and configuration
Data navigation and visualisation
Scoring/Reasoning
Reporting
Business Rule Engine
GIS and Data-Analytics Engine
PLM Core Functions
Document Vault
Data & Model Repository

Figure 6-1. Data Analytics with the sPLM System

Appendices

Appendix I: An Example for Algorithm A1 on a Simple Graph

The exact MWIS algorithms described in section 3.4 is complex. In Appendix I, we walk through Algorithm A1 in detail with a simple example in Figure 1. A simple weighted graph G shown in Figure 1 is given, with the nodes, edges, and weights shown as the figure. Note that Algorithm A2 follows a similar process, but it is returning the AMIS at each step.

All the step indexes used below are from Algorithm A1.

In step (1.1), we need to perform step (1.1.1) to (1.1.5) to find and remove nodes and update the subgraphs dictionary (SD) accordingly:

$$SD\text{: \{the } removal\ node\text{: node sets of each } connected\ component\}$$

The SD is in the format that each node removed (removed node) is the key, and node sets of each connected component in the induced subgraphs are the value of the key. The node removal process iterates until the induced subgraphs satisfy the Theorem 3-1 conditions.

Perform step (1.1.1), to find the first removed node from the input graph; we need to find a cycle basis set of the input graph. Count the occurrence of each node in the cycle basis set; the first removed node is the node that has the most occurrences. Apply Algorithm 3-1, the cycle basis algorithm, to find a cycle basis set and count the number of cycles each node belongs to. The nodes and their counts are saved in a dictionary, “occurrence_dict”: {'1': 3, '0': 3', '3': 2, '4': 2, 2': 1, '5': 1, '6': 0, '7': 0, '8': 0, '9': 0, '10': 0, '11': 0}. The occurrence of node ‘1’ and node ‘0’ both are 3; we randomly pick node ‘1’ among them. Remove node ‘1’ and the adjunct edges, the induced subgraph is illustrated as Figure 2.

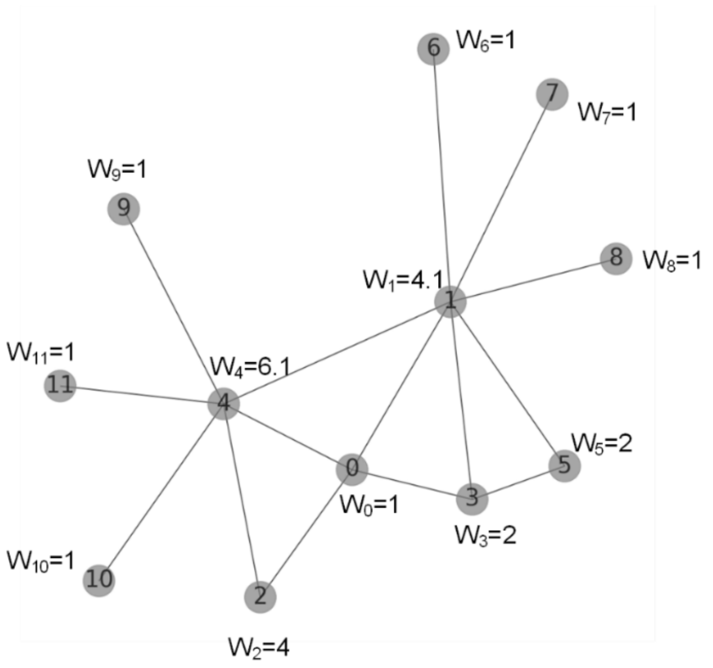


Figure 1. Simple graph for algorithm walk-through

Perform step (1.1.2), update SD with the key-value pair, SD: {'1': [{'6'}, {'7'}, {'8'}, {'4', '2', '10', '9', '11', '5', '0', '3'}]}. After removing the node ‘1’, the induced subgraph has four connected components, we use the node sets to denote these components, they are {'6'}, {'7'}, {'8'}, and {'4', '2', '10', '9', '11', '5', '0', '3'}.

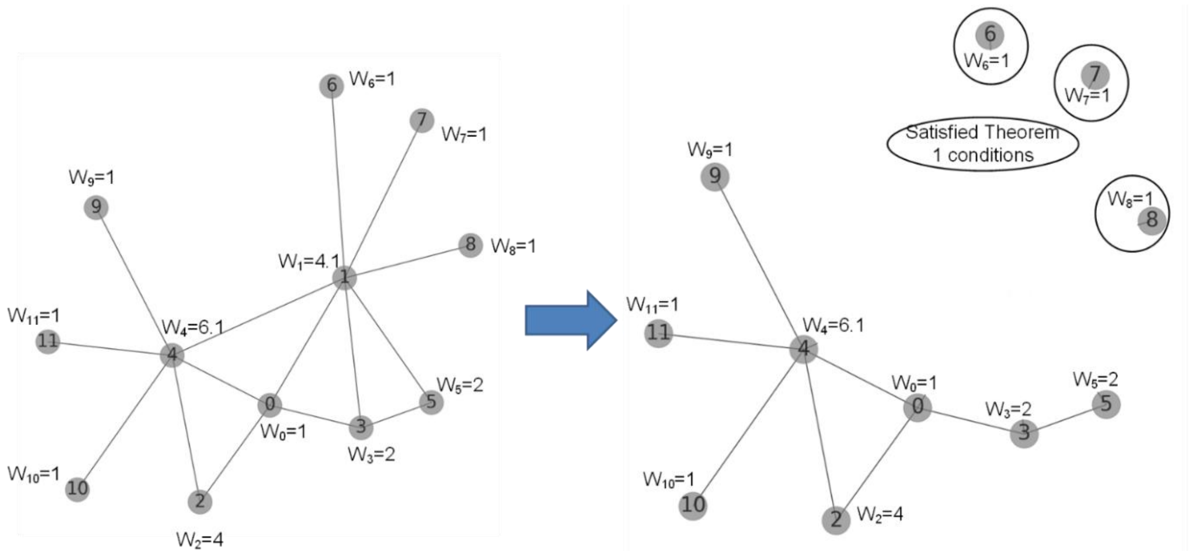


Figure 2a (left) & 3-5b (right). Remove Node ‘1’ from the Graph and the Induced Subgraph

Preform step (1.1.3), for each connected subgraph, exam whether they satisfy the Theorem 3-1 conditions. Among the four components, {'6'}, {'7'}, {'8'} satisfy the Theorem 3-1 conditions (in Figure 3-5b), but {'4', '2', '10', '9', '11', '5', '0', '3'} does not.

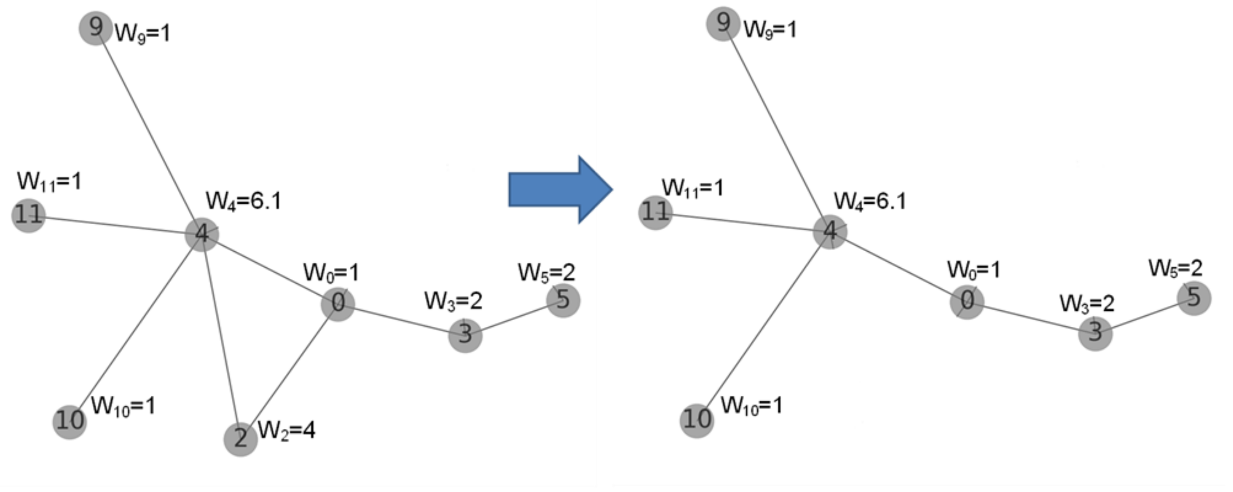


Figure 3a (left) & 3-6b (right). Remove Node ‘2’ and the Induced Subgraph

Preform step (1.1.4), the component subgraph, {'4', '2', '10', '9', '11', '5', '0', '3'}, does not satisfy the Theorem 3-1 conditions. Preform step (1.1.1), with the subgraph {'4', '2', '10', '9', '11', '5', '0', '3'}, apply Algorithm 3-1 to get current “occurrence_dict”: {'2': 1, '4': 1, '0': 1, '9': 0, '10': 0, '11': 0, '3': 0, '5': 0}. The occurrence of node ‘2’, node ‘4’ and node ‘0’ are 1, we randomly pick node ‘2’ among them. Remove node ‘2’ and the adjunct edges, the induced subgraph is illustrated as the Figure 3.

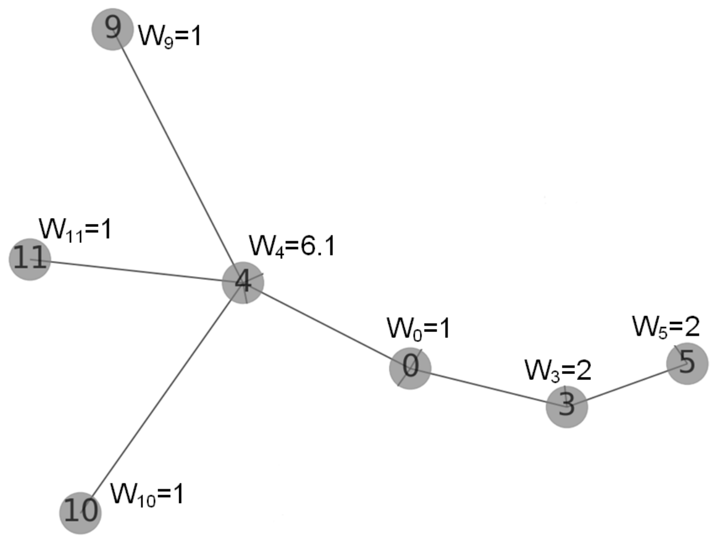


Figure 4. The induced subgraph after removing node ‘2’

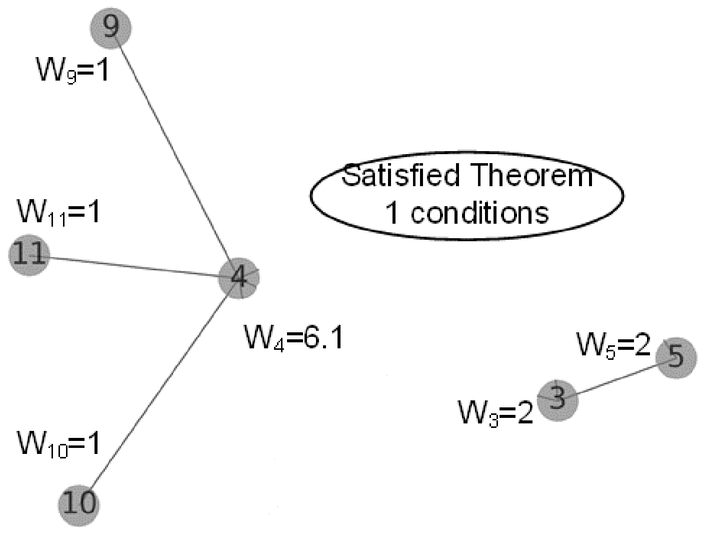


Figure 5. The induced subgraph after removing node ‘0’

Preform step (1.1.2), update SD with the key-value pair, SD: {'1': [{'6'}, {'7'}, {'8'}, {'4', '2', '10', '9', '11', '5', '0', '3'}], '2': [{'4', '10', '9', '11', '5', '0', '3'}]}. Shown as the Figure 4, the induced subgraph {'4', '10', '9', '11', '5', '0', '3'} is connected.

Preform step (1.1.3), to exam the induced subgraph. The induced subgraph {'4', '10', '9', '11', '5', '0', '3'} as-in Figure 4 does not satisfy the Theorem 3-1 conditions.

Perform step (1.1.4), by applying Algorithm 3-1, there is no cycle left in the graph {'4', '10', '9', '11', '5', '0', '3'}. Then, apply Algorithm 3-2, the diameter algorithm, this tree structure has a $diameter = 4$, which does not satisfy the Theorem 3-1 conditions. Go to step (1.1.1), with the graph {'4', '10', '9', '11', '5', '0', '3'}, apply Algorithm 3-3, the middle node algorithm, to get the middle node '0' of the tree. Remove node '0' and the adjunct edges, the induced subgraph is illustrated as Figure 5.

Preform step (1.1.2), update SD with the key-value pair, SD: {'1': [{'6'}, {'7'}, {'8'}, {'9', '2', '5', '3', '4', '10', '11', '0'}], '2': [{'9', '5', '3', '4', '10', '11', '0'}], '0': [{'5', '3'}, {'10', '9', '11', '4'}]}. After removing the node '0', the induced subgraph has two connected components, they are {'5', '3'}, and {'10', '9', '11', '4'}.

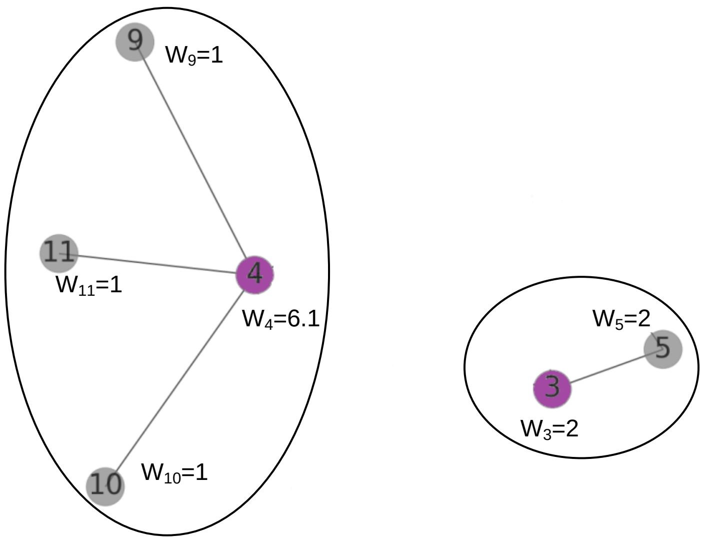


Figure 6. The Preliminary Set at the level node '0'

Perform step (1.1.3), to exam the induced subgraph. According to step (1.1.4), the two connected components in the induced subgraph both satisfy the Theorem 3-1 conditions shown in Figure 6. Jump to step (1.1.5), when all subgraphs satisfy Theorem 3-1 conditions, return the latest SD: {'1': [{'6'}, {'7'}, {'8'}, {'9', '2', '5', '3', '4', '10', '11', '0'}], '2': [{'9', '5', '3', '4', '10', '11', '0'}], '0':

[{'5', '3'}, {'10', '9', '11', '4'}]}.

All the procedures in step (1.1) for node removal are finished here.

Preform step (1.2), get the Preliminary Set from the induce subgraph according to the last key-value pair in SD. The last key-value pair in SD is: {'0': [{'5', '3'}, {'10', '9', '11', '4'}]}, indicating that at the level of node '0', there are two connected components {'5', '3'} and {'10', '9', '11', '4'}. And the Theorem 3-1 conditions are satisfied. According to Theorem 3-1, we can find the Preliminary Set for the induced subgraph with nodes {'5', '3', '10', '9', '11', '4'}. This induced subgraph is called the Preliminary Set Subgraph (PSS) at level node '0'. The Preliminary Set at the level node '0' is {4,3} with a weight total 8.1, shown as Figure 6.

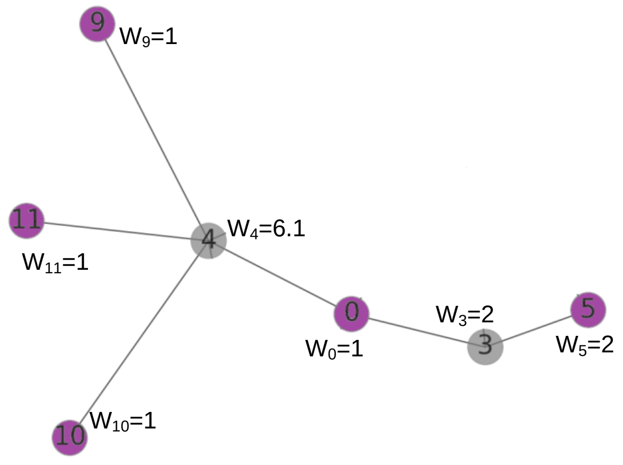


Figure 7. The Compare Set at the level node '0'

Perform step (1.3), the 'last key' is node '0'. Add node '0' to the induced subgraph (Figure 6) of step (1.2), the induced graph rolls back to Figure 4 or Figure 7. Then, follow the adding node heuristics to find the Compare Set at level node '0'. Remove the neighbors of node '0' and the adjacent edges of node '0' from Figure 7, this induced subgraph with nodes {'9', '5', '10', '11', '0'} is the Compare Set Subgraph (CSS) at the level removed node '0'. The Compare Set at the level node '0' is {'5', '0', '11', '10', '9'} with a weight total 6, shown as Figure 7.

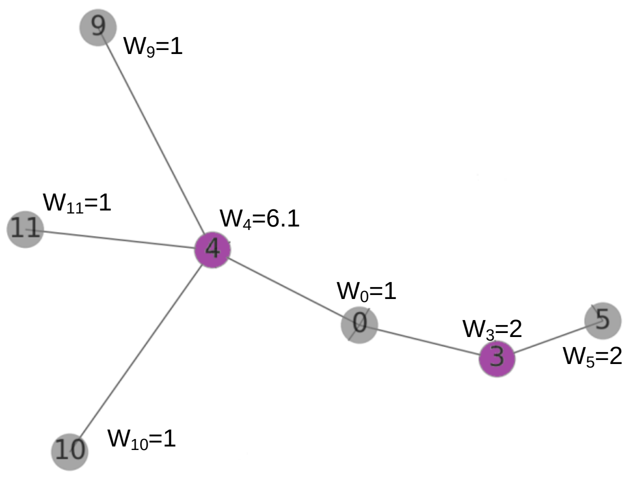


Figure 8. The MWIS at the level node ‘0’

Perform step (1.4), according to Theorem 3-2, get the set with maximum weighted total among the two sets: the Preliminary Set (Figure 6) and the Compare Set (Figure 7) at the level node ‘0’. The MWIS of the induced subgraph in Figure 8 with nodes {'0', '5', '3', '10', '9', '11', '4'} is {‘4’, ‘3’}. We can say that at level node ‘0’, the Preliminary Set is {‘4’, ‘3’} with a total weight of 8.1 as shown in Figure 8.

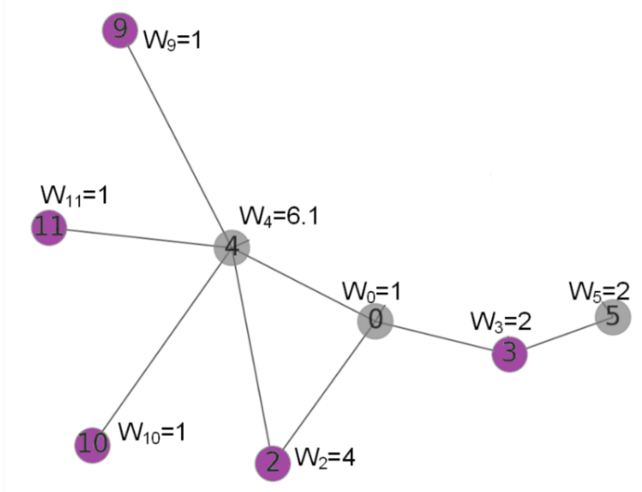


Figure 9. The Compare Set at the level node ‘2’

Preform step (1.5), update the SD as {'1': [{'6'}, {'7'}, {'8'}, {'9', '2', '5', '3', '4', '10', '11', '0'}], '2': [{'9', '5', '3', '4', '10', '11', '0'}]}. $SD \neq \emptyset$, go to step (1.2). The PSS at level node ‘2’ is the induced subgraph with nodes {'9', '5', '3', '4', '10', '11', '0'}. The Preliminary Set at level node ‘2’ is getting based on the previous step. The Preliminary Set at level node ‘2’ is the MWIS of the induced subgraph with node {'4', '10', '9', '11', '5', '0', '3’}, which is {'4', ‘3’}, and the total weights is 8.1. Preform step (1.3), the last key-value pair is the level node ‘2’. Get the Compare Set at level node ‘2’, follow the adding node heuristics. Then, the induced graph rolls back to Figure 3a. The CSS at level node ‘2’ is the induced subgraph with nodes {'9', '2', '5', '3', '10', '11'}. The Compare Set at level node ‘2’ is {'2', '3', '9', '10', '11’}. And the total weight is 9, shown as Figure 9.

Preform step (1.4), since the total weight of the Compare Set is greater than that of the Preliminary Set at level node ‘2’, according to Theorem 3-2, the induced subgraph with nodes: {'4', '2', '10', '9', '11', '5', '0', '3’} at level node ‘2’ has its MWIS as {'2', '3', '9', '10', '11’}, the total weight is 9.

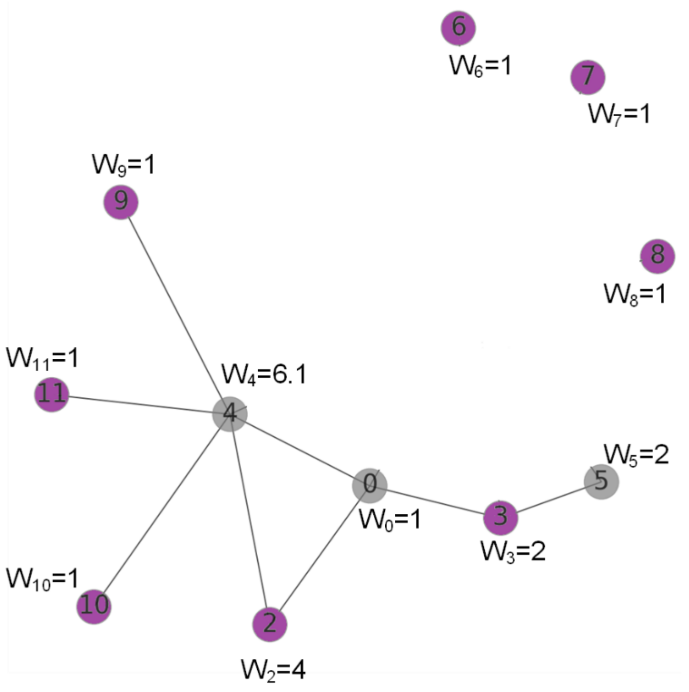


Figure 10. The Preliminary Set at the level node ‘1’

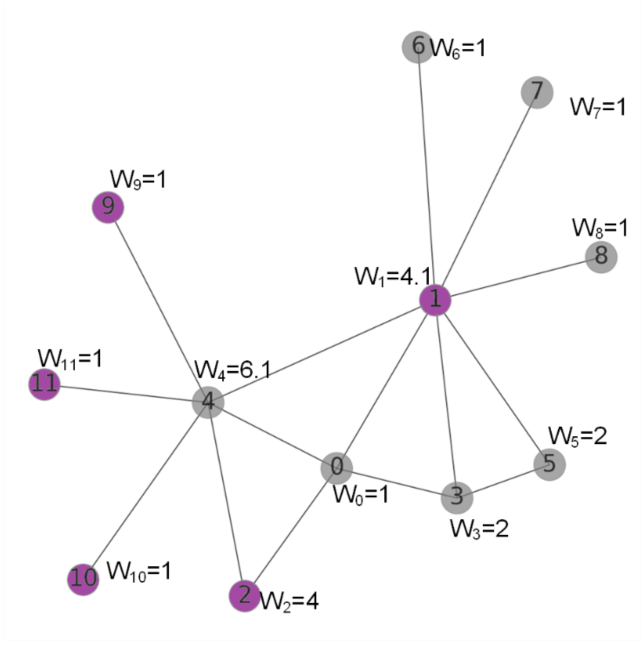


Figure 11. The Compare Set at the level node ‘1’

Perform step (1.5), update the SD as {'1': [{'6'}, {'7'}, {'8'}, {'9', '2', '5', '3', '4', '10', '11', '0'}]}. $SD \neq \emptyset$, go to step (1.2). The PSS at level node ‘1’ is the induced subgraph with nodes {'9', '5', '3', '4', '10', '11', '0', '6', '7', '8'}. The Preliminary Set at level node ‘1’ is based on the induce subgraph in the previous step. The induced subgraph at level node ‘1’ has four components: {'6'}, {'7'}, {'8'}, and {'9', '2', '5', '3', '4', '10', '11', '0'}. For the connected components, the induced subgraph with nodes {'4', '2', '10', '9', '11', '5', '0', '3’}, has its MWIS as {'2', '3', '9', '10', '11’}, the total weight is 9, same as the MWIS as level node ‘2’. According to the Theorem 3-1 and Corollary 3-1, the Preliminary Set at level node ‘1’ is the union of the MWIS of the four components with the node sets: {'6'}, {'7'}, {'8'}, and {'9', '2', '5', '3', '4', '10', '11', '0'}. The Preliminary Set at level node ‘1’ is $\{‘6’\} \cup \{‘7’\} \cup \{‘8’\} \cup \{'2', '3', '9', '10', '11’\}$, which has a total weight of 12, shown as Figure 10. Perform step (1.3), the last key-value pair is the level node ‘1’. Get the Compare Set at level node ‘1’ follow the adding node heuristics. Then, the induced graph rolls back to Figure 2a. The CSS at level node ‘1’ is the induced subgraph with nodes {'1', '9', '2', '10', '11'}. The Compare Set at level node ‘1’ is {'1', '2', '9', '10', '11’} with a total weight of 11.1, shown as Figure 11. Perform step (1.4), since the total weight of Preliminary Set is greater than that of Compare Set at level node ‘1’, according to Theorem 3-2, the induced subgraph with nodes: {'1', '6', '7', '8', '9', '2', '5', '3', '4', '10', '11', '0'} at level ‘1’ has its MWIS as {'6', '7', '8', '2', '3', '9', '10', '11’} the total weight is 12.

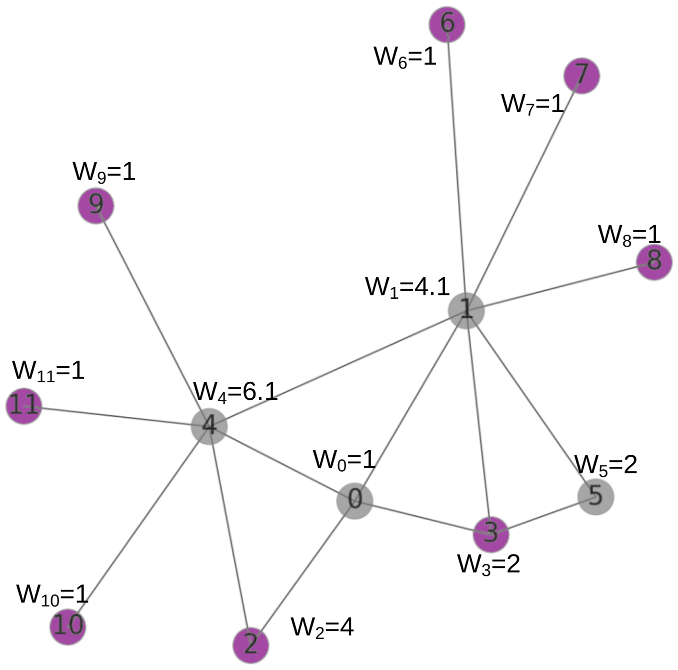


Figure 12. The MWIS of graph G

Perform step (1.5), update the SD, $SD = \emptyset$, return the MWIS of the original graph G. The MWIS is {'6', '7', '8', '2', '3', '9', '10', '11’}, the total weight is 12, shown as Figure 12.

Appendix II: Test Details of MWIS Algorithms

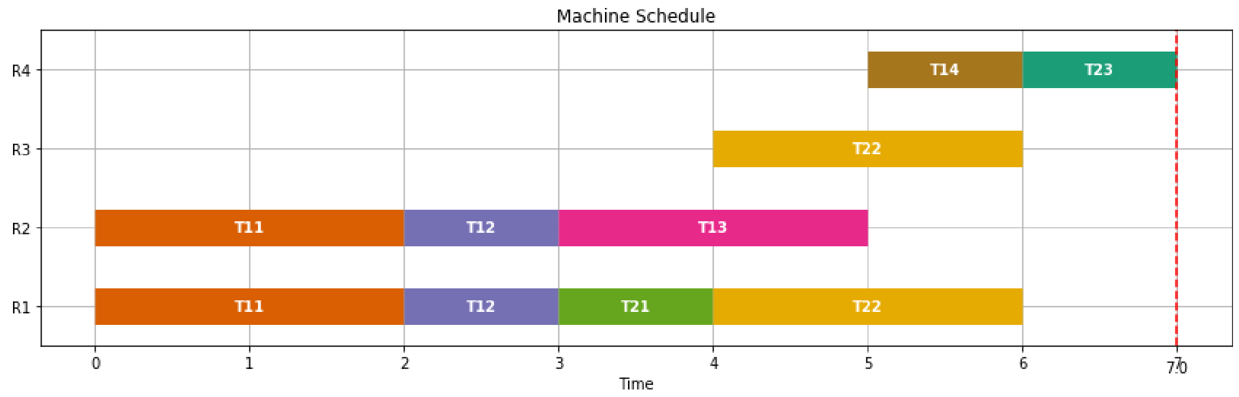
Test Info				A1		A2		A3		A4		A5		A6		A7		A8	
Test-ID	# of Edges	# of Nodes	Graph Density	Run Time	Weight Sum	Run Time	Weight Sum	Run Time	Weight Sum	Run Time	Weight Sum	Run Time	Weight Sum	Run Time	Weight Sum	Run Time	Weight Sum	Run Time	Weight Sum
1	6	5	0.6	0.118567	0.00103	0.111136	0.00103	0.000253	0.00101	0.10485	0.00103	0.10639	0.00103	0.000239	0.00101	0.162926	0.00103	0.101249	0.00103
2	9	5	0.9	0.107675	0.25001	0.087636	0.25001	0.000233	0.25	0.103948	0.25001	0.10591	0.25001	0.000207	0.25	0.106247	0.25001	0.106666	0.25001
3	29	12	0.439393939	0.393995	4.666676667	0.273826	4.666676667	0.000623	4.666676667	0.287531	4.666676667	0.293275	4.666676667	0.000625	4.583333333	0.253069	4.666676667	0.253618	4.666676667
4	29	14	0.318681319	0.485537	7.75001	0.355922	7.75001	0.001504	7.75001	0.292077	7.75001	0.382857	7.75001	0.000786	7.583333333	0.306305	7.75001	0.290889	7.75001
5	41	17	0.301470588	1.748986	3.00727	1.641865	3.00727	0.001492	3.00727	0.713053	3.00727	0.688541	3.00727	0.001111	2.090593333	0.345433	2.75729	0.397687	2.75729
6	44	17	0.323529412	0.729959	17.04168667	0.701199	17.04168667	0.002286	17.04168667	0.600235	17.04168667	0.344752	17.04168667	0.001547	16.45834333	0.469231	17.04168667	0.321118	17.04168667
7	69	17	0.507352941	0.561269	4.25001	0.368487	4.25001	0.001185	4.25	0.403378	4.25001	0.427379	4.25001	0.00099	4.25	0.405091	4.25001	0.409639	4.25001
8	52	18	0.339869281	1.448404	1.340603333	0.94847	1.340603333	0.002395	1.340603333	0.811353	1.340603333	0.627162	1.340603333	0.001794	0.590593333	0.571024	1.340603333	0.569017	1.340603333
9	6	19	0.035087719	0.556402	89.50005	0.560922	89.50005	0.002827	89.50005	0.307328	89.50005	0.310983	89.50005	0.001498	87.25004	0.424144	89.50005	0.324134	89.50005
10	0	20	0	0.001093	1639.152818	0.00279	1639.152818	0.002863	1639.152818	0.001087	1639.152818	0.001054	1639.152818	0.002053	1639.152818	0.001059	1639.152818	0.001054	1639.152818
11	8	25	0.026666667	2.112017	164.5556056	1.167152	164.5556056	0.005565	164.5556056	0.510531	164.5556056	0.503593	164.5556056	0.00294	164.5556056	0.723653	164.5556056	0.741895	164.5556056
12	50	26	0.153846154	1.581559	19.07118667	1.164716	19.07118667	0.006769	19.07118667	0.75344	19.07118667	0.716165	19.07118667	0.001923	17.73784333	0.756383	19.07118667	0.719825	19.07118667
13	101	26	0.310769231	5.463803	10.51577	4.053637	10.51577	0.004824	10.51577	0.58007	10.51577	1.428015	10.51577	0.002125	9.01578	0.642795	10.51577	0.641116	10.51577
14	115	27	0.327635328	0.401543	839.5416767	0.242558	839.5416767	0.008301	839.5416767	0.271618	839.5416767	0.194913	839.5416767	0.00288	839.5416767	0.26215	839.5416767	0.192846	839.5416767
15	147	28	0.388888889	2.431524	40.02782778	2.351423	40.02782778	0.007439	40.02782778	0.605325	40.02782778	0.507716	40.02782778	0.002718	39.11115111	0.790927	39.11115111	0.524695	39.11115111
16	155	30	0.356321839	2.160428	14.77777	1.611405	14.77777	0.004921	14.77777	0.935768	14.77777	0.821684	14.77777	0.003878	13.52776	0.975734	14.77777	0.857362	14.77777
17	135	31	0.290322581	1.605572	31.37338333	1.254355	31.37338333	0.008557	31.37338333	1.059978	31.37338333	0.697107	31.37338333	0.003946	29.54004	0.804118	31.37338333	0.704573	31.37338333
18	198	31	0.425806452	2.601461	549.3889089	2.101934	549.3889089	0.010304	547.8055656	0.723661	549.3889089	0.678604	549.3889089	0.002738	310.9444444	0.693808	549.3889089	0.777898	549.3889089
19	325	34	0.579322638	6.154325	43.13892889	4.924645	43.13892889	0.007267	43.13892889	1.771377	43.13892889	0.90644	43.13892889	0.003798	12.02777778	1.087117	41.72225222	0.761185	41.72225222
20	216	36	0.342857143	2.129672	13.87584333	1.658275	13.87584333	0.010086	13.87584333	0.818565	13.87584333	0.419063	13.87584333	0.003445	13.87584333	0.49319	13.87584333	0.367221	13.87584333
21	264	39	0.356275304	10.61476	88.92681778	9.012758	88.92681778	0.014521	88.92681778	3.646862	88.92681778	0.890331	88.92681778	0.004023	54.19858556	1.00701	88.92681778	0.841288	88.92681778
22	313	39	0.422402159	3.862964	39.62565111	3.050283	39.62565111	0.015739	39.13094333	1.032787	39.13094333	0.960322	39.62565111	0.006402	36.81481778	1.01599	39.13094333	0.954846	39.62565111
23	336	43	0.372093023	3.418986	155.50004	2.797809	155.50004	0.022145	155.50004	1.047748	155.50004	0.40617	155.50004	0.008996	105.5833633	0.457707	155.50004	0.356102	155.50004
24	388	45	0.391919192	2.597778	1309.291677	2.368349	1309.291677	0.019715	1309.291677	1.403225	1309.291677	0.517649	1309.291677	0.007383	1309.291677	0.765621	1309.291677	0.507685	1309.291677
25	425	46	0.410628019	8.003977	70.91673667	7.308244	70.91673667	0.02527	70.91673667	1.563586	70.91673667	0.675255	70.91673667	0.007505	55.66670667	0.702882	70.91673667	0.602179	70.91673667
26	583	53	0.423076923	18.92629	76.21324667	15.78386	76.21324667	0.033673	76.21324667	4.760113	76.21324667	1.667401	76.21324667	0.028344	73.71323667	2.824472	73.71323667	1.518027	73.71323667
27	465	54	0.324947589	9.57828	50.72217667	8.292436	50.72217667	0.015988	44.54351	1.105264	45.38884333	0.605086	50.72217667	0.018954	45.38884333	0.955258	45.38884333	0.50795	50.72217667
28	707	56	0.459090909	15.62693	836.4167067	12.78697	836.4167067	0.034683	836.4167067	2.639332	836.4167067	1.155839	836.4167067	0.012823	453.6388989	1.231597	836.4167067	1.106137	836.4167067
29	726	59	0.424313267	12.17281	146.37504	10.34992	146.37504	0.036667	124.0417067	1.574886	146.37504	1.028119	146.37504	0.014819	95.95840333	1.982005	124.0417067	0.982974	146.37504
30	726	59	0.424313267	11.86624	1510.37504	10.29127	1510.37504	0.033442	1457.041707	1.685658	1510.37504	1.029224	1510.37504	0.014331	1227.458403	1.065756	1457.041707	0.99773	1510.37504
31	723	61	0.395081967	25.16087	126.5310522	22.26743	126.5310522	0.085729	126.5310522	6.561642	126.5310522	2.307274	126.5310522	0.041315	123.3643756	0.176591	123.3643756	1.650675	123.3643756
32	840	63	0.430107527	45.89679	86.97227222	42.46756	86.97227222	0.051448	86.97227222	5.779079	86.97227222	2.353676	86.97227222	0.017616	63.97223222	3.711723	85.05559556	2.361078	85.05559556
33	980	67	0.443238354	16.00381	148.5972622	14.50707	148.5972622	0.054451	143.2639289	3.510512	148.5972622	1.769027	148.5972622	0.022087	98.51395889	1.745536	124.4861511	1.450741	148.5972622
34	1238	79	0.401817592	108.7613	146.75005	93.71785	146.75005	0.074164	146.75005	4.028621	146.75005	2.699751	146.75005	0.030016	38.77783778	2.930075	145.75005	2.488386	145.75005
35	1387	89	0.354187947	106.4795	237.4167367	94.94339	237.4167367	0.107995	237.4167367	4.876775	237.4167367	2.619159	237.4167367	0.050076	216.25009	2.955594	216.25009	2.067732	237.4167367
36	1387	89	0.354187947	104.5018	156.7474867	94.51976	156.7474867	0.094051	156.7474867	5.352133	156.7474867	2.602482	156.7474867	0.042201	91.73248667	2.735964	124.7384867	2.070066	156.7474867
37	1387	89	0.354187947	104.5936	2579.166737	95.38101	2579.166737	0.088363	2579.166737	4.599551	2579.166737	2.542564	2579.166737	0.041883	2087.916677	2.835716	2277.166737	2.062046	2579.166737
38	2184	110	0.364303586	460.6811	259.8334333	457.9928	259.8334333	0.146865	227.5834933	10.13942	227.5834933	5.899711	259.8334333	0.097114	241.3334933	7.294095	241.3334933	4.462351	254.0834333
39	2184	110	0.364303586	487.6465	2670.083433	484.6684	2670.083433	0.157923	2490.583493	9.573596	2658.916767	5.832959	2670.083433	0.081097	1988.083373	5.909477	2113.333483	4.697975	2431.916817
40	3108	126	0.394666667	978.4428	262.2778778	924.2892	262.2778778	0.22231	226.7223322	13.62323	262.2778778	9.455681	262.2778778	0.168643	201.6946044	11.76862	244.3334933	7.541868	262.2778778
41	3108	126	0.394666667	967.6664	2672.527878	927.3733	2672.527878	0.235084	2672.527878	17.15051	2672.527878	9.302291	2672.527878	0.176047	1885.527818	10.97006	2498.583493	7.475677	2672.527878
42	4718	161	0.366304348	13046.56	175.4459711	12705.98	175.4459711	0.274509	175.4459711	36.28218	175.4459711	28.67411	175.4459711	0.401469	90.89372333	34.98096	142.4607367	22.34016	170.5250444
43	4718	161	0.366304348	12041.21	260.1112211	11761.95	260.1112211	0.272659	260.1112211	43.85244	260.1112211	25.79004	260.1112211	0.395887	161.8890189	32.65466	202.8612811	21.44091	250.9445444

Appendix III: PPS Test Instances

I. Testing 1:

Job #1: $\frac{T_{11}[(R_1,R_2,R_3)_2]}{2TS} \rightarrow \frac{T_{12}[(R_1,R_2,R_3)_2]}{1TS} \rightarrow \frac{T_{13}[(R_1,R_2,R_3)_1]}{2TS} \rightarrow \frac{T_{14}[(R_4)_1]}{1TS}$

Job #2: $\frac{T_{21}[(R_1,R_2,R_3)_1]}{1TS} \rightarrow \frac{T_{22}[(R_1)_1\text{and}(R_2,R_3)_1]}{2TS} \rightarrow \frac{T_{23}[(R_4)_1]}{1TS}$

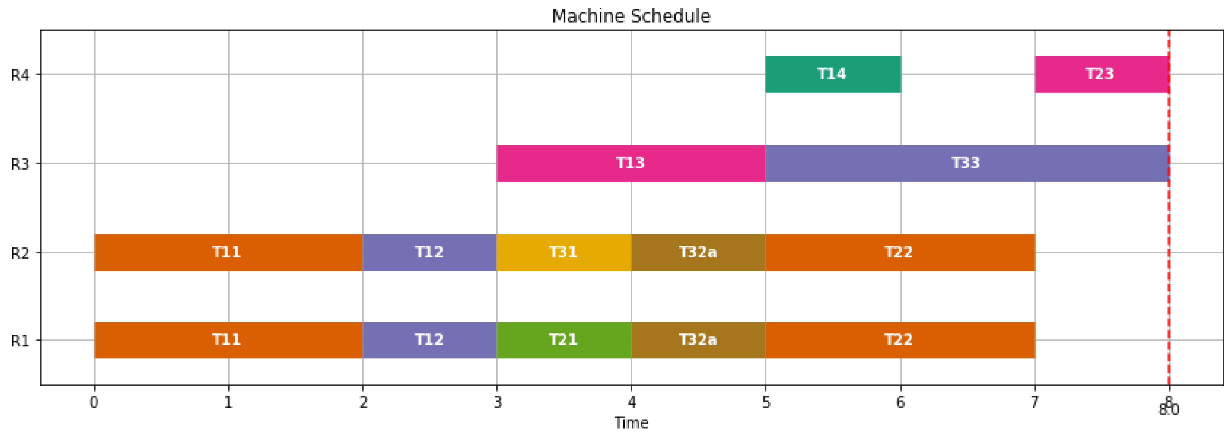


II. Testing 2

Job #1: $\frac{T_{11}[(R_1,R_2,R_3)_2]}{2TS} \rightarrow \frac{T_{12}[(R_1,R_2,R_3)_2]}{1TS} \rightarrow \frac{T_{13}[(R_1,R_2,R_3)_1]}{2TS} \rightarrow \frac{T_{14}[(R_4)_1]}{1TS}$

Job #2: $\frac{T_{21}[(R_1,R_2,R_3)_1]}{1TS} \rightarrow \frac{T_{22}[(R_1)_1\text{and}(R_2,R_3)_1]}{2TS} \rightarrow \frac{T_{23}[(R_4)_1]}{1TS}$

Job #3: $\frac{T_{31}[(R_1,R_2)_1]}{1TS} \rightarrow \begin{pmatrix} \frac{T_{32a}[(R_1,R_2)_2]}{1TS} \\ \frac{T_{32b}[(R_3)_1]}{2TS} \end{pmatrix}^1 \rightarrow \frac{T_{33}[(R_3,R_4)_1]}{3TS}$

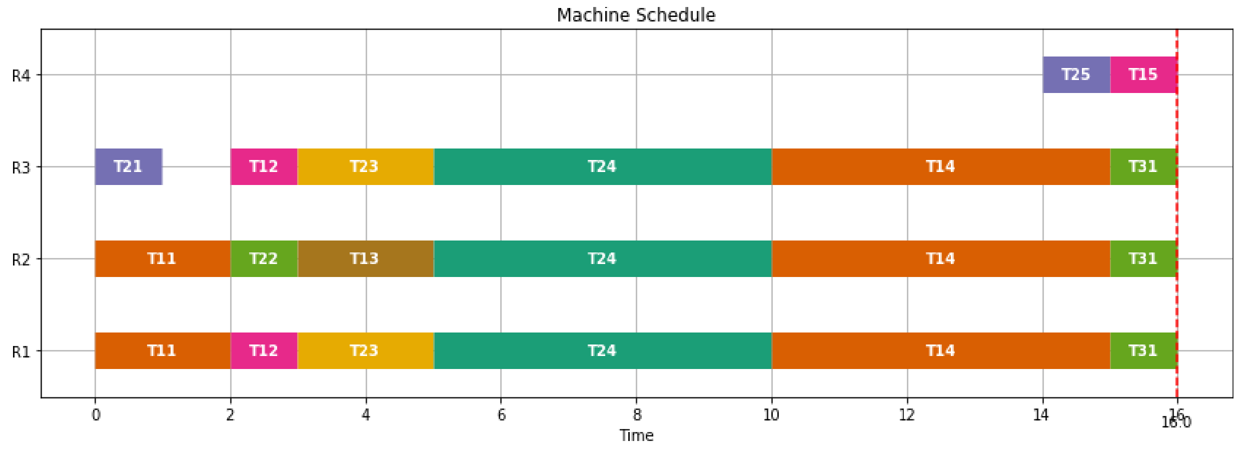


III. Testing 3

Job #1: $\frac{T_{11}[(R_1,R_2,R_3)_2]}{2TS} \rightarrow \frac{T_{12}[(R_1,R_2,R_3)_2]}{1TS} \rightarrow \frac{T_{13}[(R_1,R_2,R_3)_1]}{2TS} \rightarrow \frac{T_{14}[(R_1,R_2,R_3)_3]}{5TS} \rightarrow \frac{T_{15}[(R_4)_1]}{1TS}$

Job #2: $\frac{T_{21}[(R_1,R_2,R_3)_1]}{1TS} \rightarrow \frac{T_{22}[(R_1,R_2)_1]}{1TS} \rightarrow \frac{T_{23}[(R_1)_1\text{and}(R_2,R_3)_1]}{2TS} \rightarrow \frac{T_{24}[(R_1,R_2,R_3)_3]}{5TS} \rightarrow \frac{T_{25}[(R_4)_1]}{1TS}$

Job #3: $\frac{T_{31}[(R_1,R_2,R_3)_3]}{1TS}$

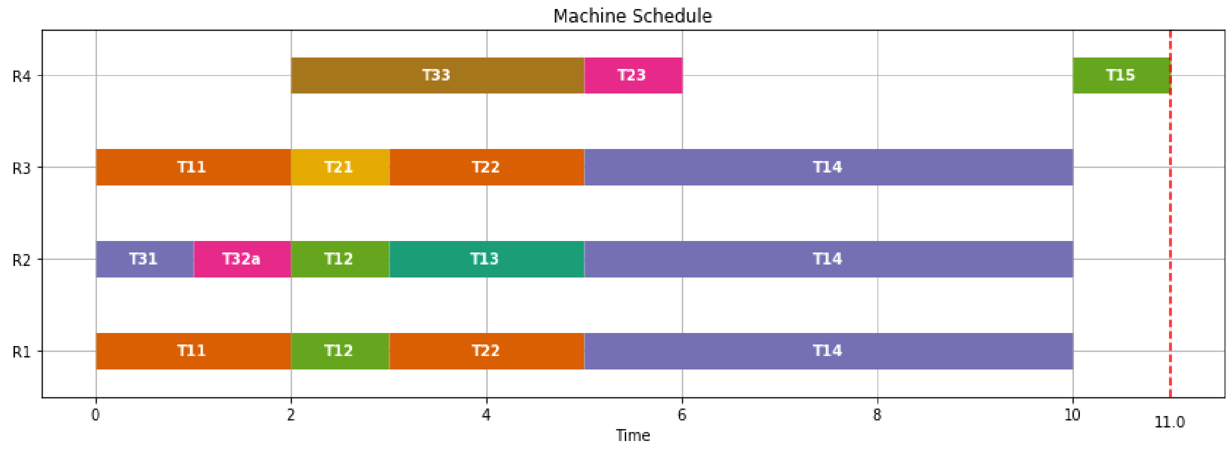


IV. Testing 4

Job #1: $\frac{T_{11}[(R_1,R_2,R_3)_2]}{2TS} \rightarrow \frac{T_{12}[(R_1,R_2,R_3)_2]}{1TS} \rightarrow \frac{T_{13}[(R_1,R_2,R_3)_1]}{2TS} \rightarrow \frac{T_{14}[(R_1,R_2,R_3)_3]}{5TS} \rightarrow \frac{T_{15}[(R_4)_1]}{1TS}$

Job #2: $\frac{T_{21}[(R_1,R_2,R_3)_1]}{1TS} \rightarrow \frac{T_{22}[(R_1)_1\text{and}(R_2,R_3)_1]}{2TS} \rightarrow \frac{T_{23}[(R_4)_1]}{1TS}$

Job #3: $\frac{T_{31}[(R_1,R_2)_1]}{1TS} \rightarrow \begin{pmatrix} \frac{T_{32a}[(R_1,R_2)_1]}{1TS} \\ \frac{T_{32b}[(R_3)_1]}{2TS} \end{pmatrix}^1 \rightarrow \frac{T_{33}[(R_3,R_4)_1]}{3TS}$



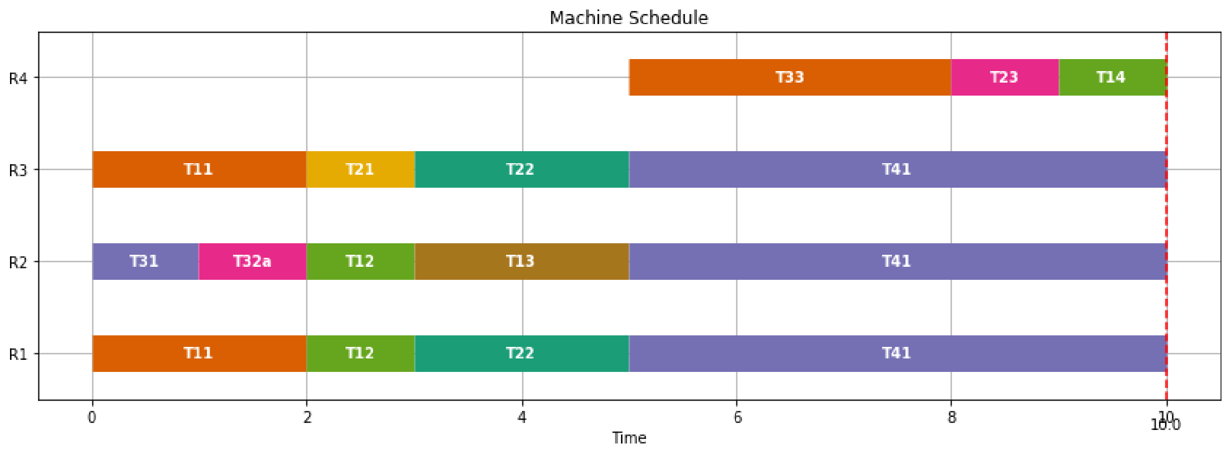
V. Testing 5

Job #1: $\frac{T_{11}[(R_1,R_2,R_3)_2]}{2TS} \rightarrow \frac{T_{12}[(R_1,R_2,R_3)_2]}{1TS} \rightarrow \frac{T_{13}[(R_1,R_2,R_3)_1]}{2TS} \rightarrow \frac{T_{14}[(R_4)_1]}{1TS}$

Job #2: $\frac{T_{21}[(R_1,R_2,R_3)_1]}{1TS} \rightarrow \frac{T_{22}[(R_1)_1\text{and}(R_2,R_3)_1]}{2TS} \rightarrow \frac{T_{23}[(R_4)_1]}{1TS}$

Job #3: $\frac{T_{31}[(R_1,R_2)_1]}{1TS} \rightarrow \begin{pmatrix} \frac{T_{32a}[(R_1,R_2)_1]}{1TS} \\ \frac{T_{32b}[(R_3)_1]}{2TS} \end{pmatrix}^1 \rightarrow \frac{T_{33}[(R_3,R_4)_1]}{3TS}$

Job #4: $\frac{T_{41}[(R_1,R_2,R_3)_3]}{5TS}$



VI. Testing 6

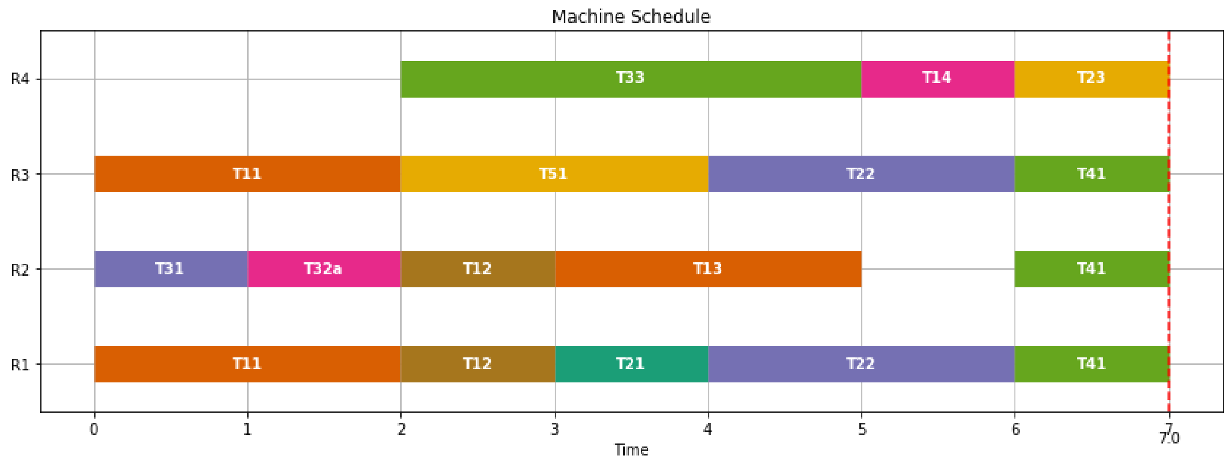
Job #1: $\frac{T_{11}[(R_1,R_2,R_3)_2]}{2TS} \rightarrow \frac{T_{12}[(R_1,R_2,R_3)_2]}{1TS} \rightarrow \frac{T_{13}[(R_1,R_2,R_3)_1]}{2TS} \rightarrow \frac{T_{14}[(R_4)_1]}{1TS}$

Job #2: $\frac{T_{21}[(R_1,R_2,R_3)_1]}{1TS} \rightarrow \frac{T_{22}[(R_1)_1\text{and}(R_2,R_3)_1]}{2TS} \rightarrow \frac{T_{23}[(R_4)_1]}{1TS}$

Job #3: $\frac{T_{31}[(R_1,R_2)_1]}{1TS} \rightarrow \begin{pmatrix} \frac{T_{32a}[(R_1,R_2)_1]}{1TS} \\ \frac{T_{32b}[(R_3)_1]}{2TS} \end{pmatrix}^1 \rightarrow \frac{T_{33}[(R_3,R_4)_1]}{3TS}$

Job #4: $\frac{T_{41}[(R_1,R_2,R_3)_3]}{1TS}$

Job #5: $\frac{T_{51}[(R_1,R_2,R_3)_1]}{2TS}$



VII. Testing 7

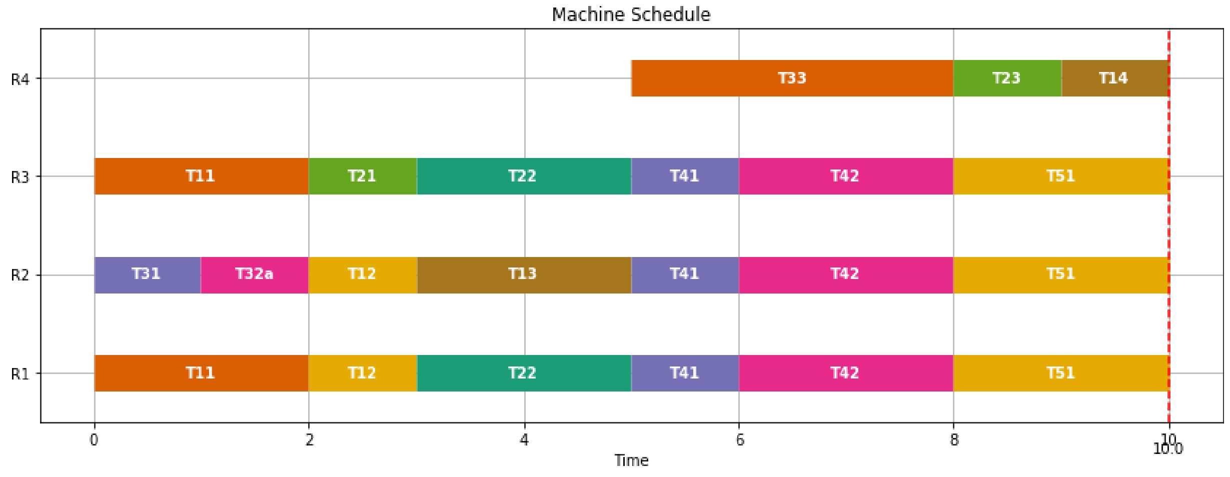
Job #1: $\frac{T_{11}[(R_1,R_2,R_3)_2]}{2TS} \rightarrow \frac{T_{12}[(R_1,R_2,R_3)_2]}{1TS} \rightarrow \frac{T_{13}[(R_1,R_2,R_3)_1]}{2TS} \rightarrow \frac{T_{14}[(R_4)_1]}{1TS}$

Job #2: $\frac{T_{21}[(R_1,R_2,R_3)_1]}{1TS} \rightarrow \frac{T_{22}[(R_1)_1\text{and}(R_2,R_3)_1]}{2TS} \rightarrow \frac{T_{23}[(R_4)_1]}{1TS}$

Job #3: $\frac{T_{31}[(R_1,R_2)_1]}{1TS} \rightarrow \begin{pmatrix} \frac{T_{32a}[(R_1,R_2)_2]}{1TS} \\ \frac{T_{32b}[(R_3)_1]}{2TS} \end{pmatrix}^1 \rightarrow \frac{T_{33}[(R_3,R_4)_1]}{3TS}$

Job #4: $\frac{T_{41}[(R_1,R_2,R_3)_3]}{1TS} \rightarrow \frac{T_{42}[(R_1,R_2,R_3)_3]}{2TS}$

Job #5: $\frac{T_{51}[(R_1,R_2,R_3)_3]}{2TS}$



VIII. Testing 8

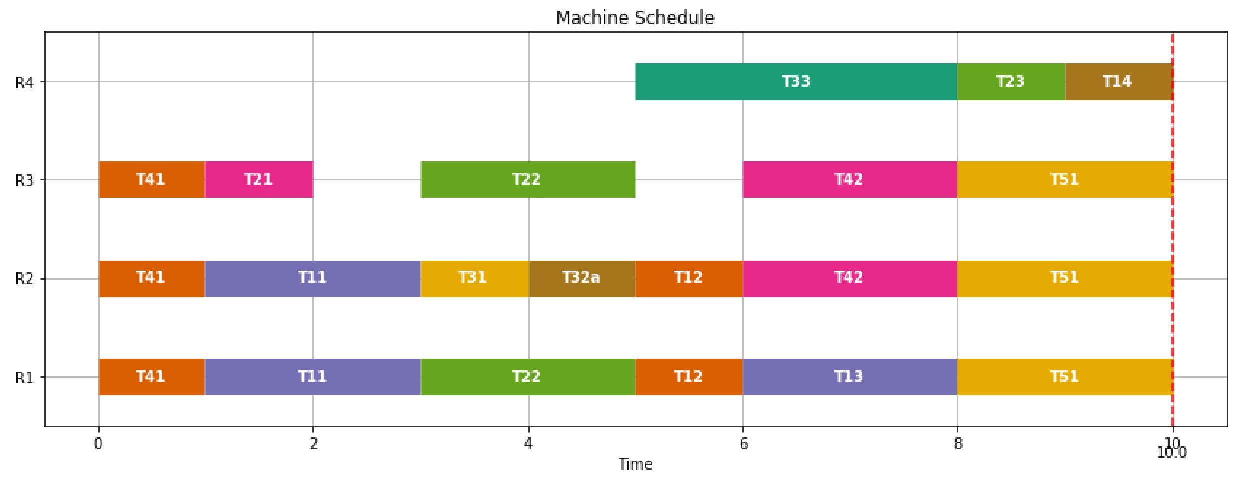
Job #1: $\frac{T_{11}[(R_1,R_2,R_3)_2]}{2TS} \rightarrow \frac{T_{12}[(R_1,R_2,R_3)_2]}{1TS} \rightarrow \frac{T_{13}[(R_1,R_2,R_3)_1]}{2TS} \rightarrow \frac{T_{14}[(R_4)_1]}{1TS}$

Job #2: $\frac{T_{21}[(R_1,R_2,R_3)_1]}{1TS} \rightarrow \frac{T_{22}[(R_1)_1\text{and}(R_2,R_3)_1]}{2TS} \rightarrow \frac{T_{23}[(R_4)_1]}{1TS}$

Job #3: $\frac{T_{31}[(R_1,R_2)_1]}{1TS} \rightarrow \begin{pmatrix} \frac{T_{32a}[(R_1,R_2)_2]}{1TS} \\ \frac{T_{32b}[(R_3)_1]}{2TS} \end{pmatrix}^1 \rightarrow \frac{T_{33}[(R_3,R_4)_1]}{3TS}$

Job #4: $\frac{T_{41}[(R_1,R_2,R_3)_3]}{1TS} \rightarrow \frac{T_{42}[(R_1,R_2,R_3)_2]}{2TS}$

Job #5: $\frac{T_{51}[(R_1,R_2,R_3)_3]}{2TS}$



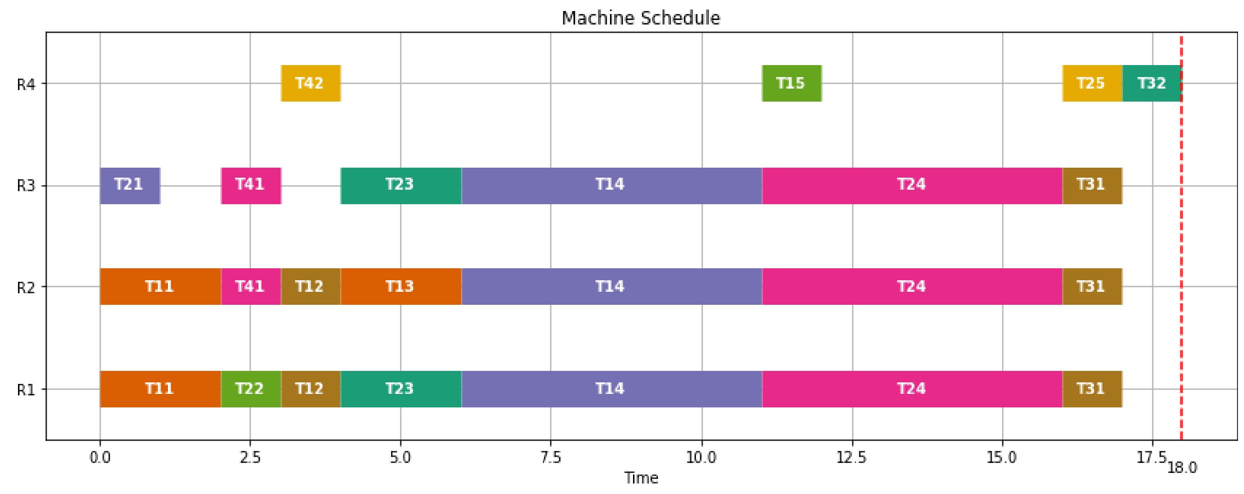
IX. Testing 9

Job #1: $\frac{T_{11}[(R_1,R_2,R_3)_2]}{2TS} \rightarrow \frac{T_{12}[(R_1,R_2,R_3)_2]}{1TS} \rightarrow \frac{T_{13}[(R_1,R_2,R_3)_1]}{2TS} \rightarrow \frac{T_{14}[(R_1,R_2,R_3)_3]}{5TS} \rightarrow \frac{T_{15}[(R_4)_1]}{1TS}$

Job #2: $\frac{T_{21}[(R_1,R_2,R_3)_1]}{1TS} \rightarrow \frac{T_{22}[(R_1,R_2)_1]}{1TS} \rightarrow \frac{T_{23}[(R_1)_1\text{and}(R_2,R_3)_1]}{2TS} \rightarrow \frac{T_{24}[(R_1,R_2,R_3)_3]}{5TS} \rightarrow \frac{T_{25}[(R_4)_1]}{1TS}$

Job #3: $\frac{T_{31}[(R_1,R_2,R_3)_3]}{1TS} \rightarrow \frac{T_{32}[(R_4)_1]}{1TS}$

Job #4: $\frac{T_{41}[(R_3)_1\text{and}(R_1,R_2)_1]}{1TS} \rightarrow \frac{T_{42}[(R_4)_1]}{1TS}$



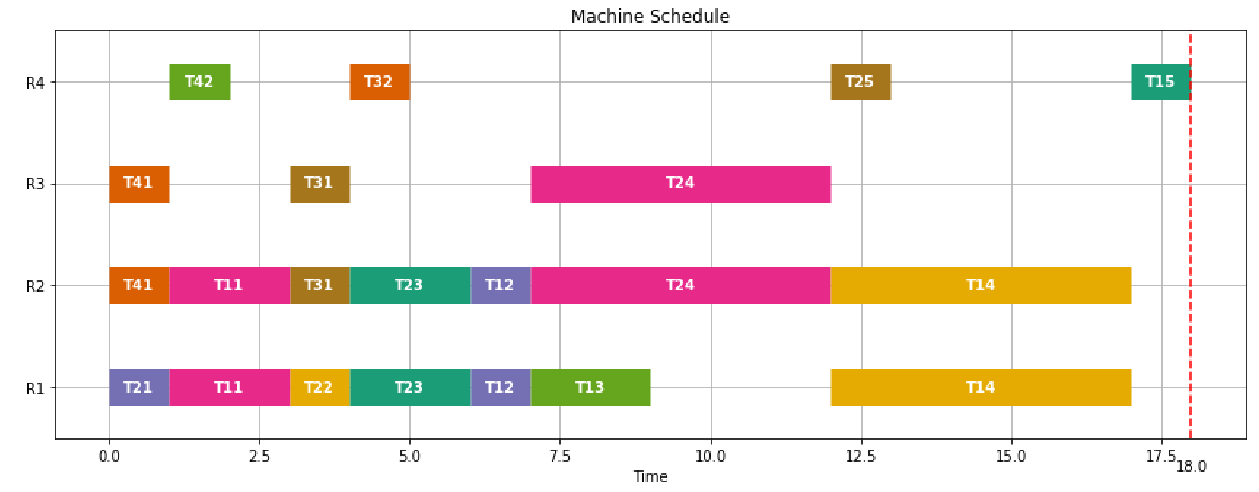
X. Testing 10

Job #1: $\frac{T_{11}[(R_1,R_2,R_3)_2]}{2TS} \rightarrow \frac{T_{12}[(R_1,R_2,R_3)_2]}{1TS} \rightarrow \frac{T_{13}[(R_1,R_2,R_3)_1]}{2TS} \rightarrow \frac{T_{14}[(R_1,R_2,R_3)_2]}{5TS} \rightarrow \frac{T_{15}[(R_4)_1]}{1TS}$

Job #2: $\frac{T_{21}[(R_1,R_2,R_3)_1]}{1TS} \rightarrow \frac{T_{22}[(R_1,R_2)_1]}{1TS} \rightarrow \frac{T_{23}[(R_1)_1\text{and}(R_2,R_3)_1]}{2TS} \rightarrow \frac{T_{24}[(R_1,R_2,R_3)_2]}{5TS} \rightarrow \frac{T_{25}[(R_4)_1]}{1TS}$

Job #3: $\frac{T_{31}[(R_1,R_2,R_3)_2]}{1TS} \rightarrow \frac{T_{32}[(R_4)_1]}{1TS}$

Job #4: $\frac{T_{41}[(R_3)_1\text{and}(R_1,R_2)_1]}{1TS} \rightarrow \frac{T_{42}[(R_4)_1]}{1TS}$



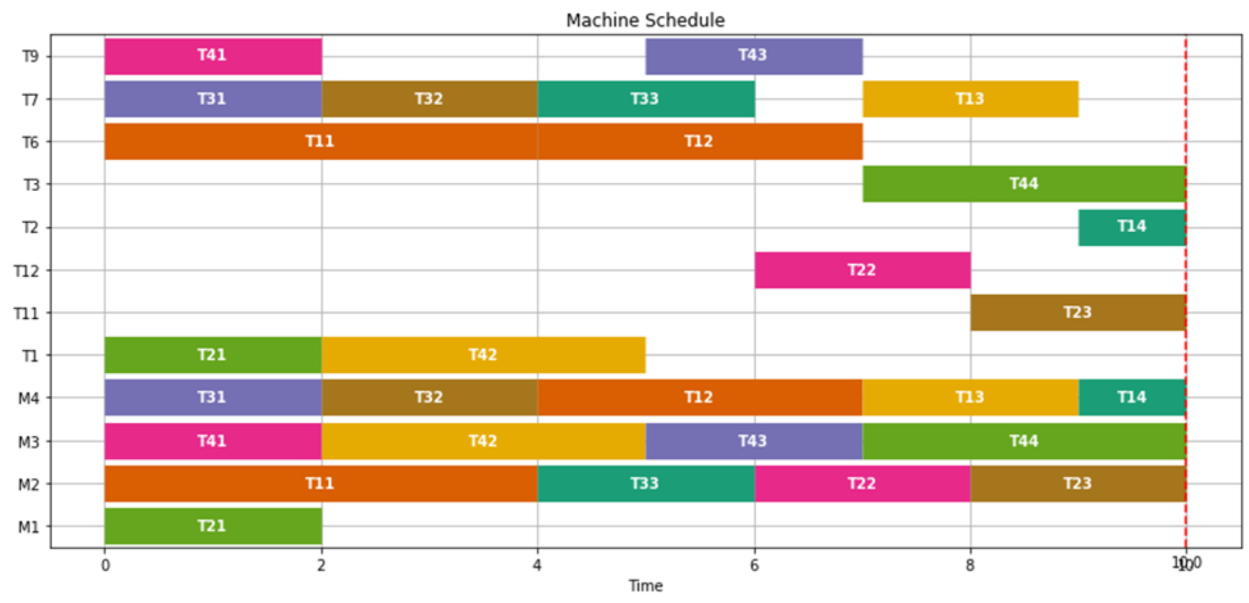
XI. Testing 11

Job #1: $\left(\begin{array}{c}\frac{T_{1,1a}[(M_2,M_3)_1 \text{ and } (T_6,T_7)_1]}{4TS}\\ \frac{T_{1,1b}[(M_4)_1 \text{ and } (T_6,T_7)_1]}{3TS}\end{array}\right)^1 \rightarrow \left(\begin{array}{c}\frac{T_{1,2a}[(M_2,M_3)_1 \text{ and } (T_6,T_7)_1]}{4TS}\\ \frac{T_{1,2b}[(M_4)_1 \text{ and } (T_6,T_7)_1]}{3TS}\end{array}\right)^1 \rightarrow \left(\frac{T_{1,3a}[(M_2,M_3,M_4)_1 \text{ and } (T_6,T_7)_1]}{2TS}\right)^1 \rightarrow \left(\begin{array}{c}\frac{T_{1,4a}[(M_1)_1 \text{ and } (T_2)_1]}{2TS}\\ \frac{T_{1,4b}[(M_2,M_3,M_4)_1 \text{ and } (T_2)_1]}{1TS}\end{array}\right)^1$

Job #2: $\left(\begin{array}{c}\frac{T_{2,1b}[(M_1)_1 \text{ and } (T_1)_1]}{2TS}\\ \frac{T_{2,1a}[(M_2,M_3,M_4)_1 \text{ and } (T_1)_1]}{1TS}\end{array}\right)^1 \rightarrow \left(\frac{T_{2,2a}[(M_2,M_3,M_4)_1 \text{ and } (T_{12})_1]}{2TS}\right)^1 \rightarrow \left(\frac{T_{2,3a}[(M_2,M_3,M_4)_1 \text{ and } (T_6,T_7,T_{11})_1]}{2TS}\right)^1$

Job #3: $\left(\frac{T_{3,1a}[(M_2,M_3,M_4)_1 \text{ and } (T_7,T_8)_1]}{2TS}\right)^1 \rightarrow \left(\frac{T_{3,2a}[(M_2,M_3,M_4)_1 \text{ and } (T_7,T_8)_1]}{2TS}\right)^1 \rightarrow \left(\frac{T_{3,3a}[(M_2,M_3,M_4)_1 \text{ and } (T_7,T_8)_1]}{2TS}\right)^1$

Job #4: $\left(\begin{array}{c}\frac{T_{4,1a}[(M_2)_1 \text{ and } (T_9,T_{10})_1]}{3TS}\\ \frac{T_{4,1b}[(M_3)_1 \text{ and } (T_9,T_{10})_1]}{2TS}\end{array}\right)^1 \rightarrow \left(\frac{T_{4,2a}[(M_2,M_3)_1 \text{ and } (T_1,T_3)_1]}{3TS}\right)^1 \rightarrow \left(\frac{T_{4,3a}[(M_2,M_3)_1 \text{ and } (T_6,T_9)_1]}{2TS}\right)^1 \rightarrow \left(\begin{array}{c}\frac{T_{4,4a}[(M_2)_1 \text{ and } (T_3)_1]}{2TS}\\ \frac{T_{4,4b}[(M_3)_1 \text{ and } (T_3)_1]}{3TS}\end{array}\right)^1$



XII. Testing 12

In test instance 12, the machine M_5 is loaded as conflict with other resources.

Job #1: $\begin{pmatrix} \frac{T_{11a}[(M_2, M_3)_1 \text{ and } (T_6,T_7,T_8)_1]}{3TS} \\ \frac{T_{11b}[(M_4)_1 \text{ and } (T_6,T_7,T_8)_1]}{2TS} \end{pmatrix}^1 \rightarrow \begin{pmatrix} \frac{T_{12a}[(M_2, M_3)_1 \text{ and } (T_6,T_7,T_8)_1]}{3TS} \\ \frac{T_{12b}[(M_4)_1 \text{ and } (T_6,T_7,T_8)_1]}{2TS} \end{pmatrix}^1 \rightarrow \begin{pmatrix} \frac{T_{13a}[(M_2, M_3)_1 \text{ and } (T_6,T_7,T_8)_1]}{2TS} \\ \frac{T_{13b}[(M_4)_1 \text{ and } (T_6,T_7,T_8)_1]}{1TS} \end{pmatrix}^1 \rightarrow$

$\begin{pmatrix} \frac{T_{14a}[(M_2, M_3)_1 \text{ and } (T_6,T_7)_1]}{3TS} \\ \frac{T_{14b}[(M_4)_1 \text{ and } (T_6,T_7)_1]}{2TS} \end{pmatrix}^1 \rightarrow \left(\frac{T_{15a}[(M_2, M_3,M_4)_1 \text{ and } (T_7,T_8)_1]}{1TS}\right)^1 \rightarrow \left(\frac{T_{16a}[(M_2, M_3,M_4)_1 \text{ and } (T_7,T_8)_1]}{1TS}\right)^1 \rightarrow$

$\begin{pmatrix} \frac{T_{17a}[(M_2, M_3)_1 \text{ and } (T_6,T_7)_1]}{3TS} \\ \frac{T_{17b}[(M_4)_1 \text{ and } (T_6,T_7)_1]}{2TS} \end{pmatrix}^1 \rightarrow \left(\frac{T_{18a}[(M_1,M_2, M_3,M_4)_1 \text{ and } (T_2)_1]}{1TS}\right)^1 \rightarrow \left(\frac{T_{19a}[(M_2, M_3,M_4)_1 \text{ and } (T_9)_1]}{2TS}\right)^1 \rightarrow$

$\begin{pmatrix} \frac{T_{110a}[(M_1,M_2, M_3)_1 \text{ and } (T_2,T_3,T_4)_1]}{3TS} \\ \frac{T_{110b}[(M_4)_1 \text{ and } (T_2,T_3,T_4)_1]}{2TS} \end{pmatrix}^1 \rightarrow \left(\frac{T_{111a}[(M_2, M_3,M_4)_1 \text{ and } (T_9)_1]}{2TS}\right)^1 \rightarrow \left(\frac{T_{112a}[(M_2,M_3, M_4,M_5)_1 \text{ and } (T_{10})_1]}{2TS}\right)^1 \rightarrow$

$\left(\frac{T_{113a}[(M_1,M_2, M_3,M_4)_1 \text{ and } (T_1)_1]}{2TS}\right)^1 \rightarrow \begin{pmatrix} \frac{T_{114a}[(M_2, M_3)_1 \text{ and } (T_5)_1]}{2TS} \\ \frac{T_{114b}[(M_4)_1 \text{ and } (T_5)_1]}{1TS} \end{pmatrix}^1 \rightarrow \begin{pmatrix} \frac{T_{115a}[(M_2, M_3)_1 \text{ and } (T_7,T_8)_1]}{2TS} \\ \frac{T_{115b}[(M_4)_1 \text{ and } (T_7,T_8)_1]}{1TS} \end{pmatrix}^1 \rightarrow$

$\begin{pmatrix} \frac{T_{116a}[(M_1,M_2, M_3)_1 \text{ and } (T_2,T_3,T_4)_1]}{2TS} \\ \frac{T_{116b}[(M_4)_1 \text{ and } (T_2,T_3,T_4)_1]}{1TS} \end{pmatrix}^1 \rightarrow \left(\frac{T_{117a}[(M_2,M_3, M_4)_1 \text{ and } (T_9)_1]}{1TS}\right)^1 \rightarrow \left(\frac{T_{118a}[(M_2,M_3, M_4,M_5)_1 \text{ and } (T_{10})_1]}{1TS}\right)^1 \rightarrow$

$\left(\frac{T_{119a}[(M_2,M_3, M_4)_1 \text{ and } (T_9)_1]}{1TS}\right)^1 \rightarrow \left(\frac{T_{112a}[(M_2,M_3, M_4,M_5)_1 \text{ and } (T_{10})_1]}{1TS}\right)^1$

Job #2: $\left(\frac{T_{21a}[(M_2, M_3,M_4)_1 \text{ and } (T_6,T_7,T_8)_1]}{1TS}\right)^1 \rightarrow \left(\begin{matrix}\frac{T_{22a}[(M_1)_1 \text{ and } (T_6,T_7,T_8)_1]}{3TS}\\ \frac{T_{22b}[(M_2, M_3,M_4)_1 \text{ and } (T_6,T_7,T_8)_1]}{2TS}\end{matrix}\right)^1 \rightarrow \left(\frac{T_{23a}[(M_2, M_3,M_4)_1 \text{ and } (T_1)_1]}{1TS}\right)^1 \rightarrow$

$\left(\begin{matrix}\frac{T_{24a}[(M_2, M_3)_1 \text{ and } (T_5,T_6,T_{11})_1]}{2TS}\\ \frac{T_{24b}[(M_4)_1 \text{ and } (T_5,T_6,T_{11})_1]}{1TS}\end{matrix}\right)^1 \rightarrow \left(\frac{T_{25a}[(M_1,M_2, M_3,M_4)_1 \text{ and } (T_2)_1]}{2TS}\right)^1 \rightarrow \left(\frac{T_{26a}[(M_2, M_3,M_4)_1 \text{ and } (T_6,T_7,T_8)_1]}{1TS}\right)^1 \rightarrow$

$\left(\frac{T_{27a}[(M_2, M_3,M_4)_1 \text{ and } (T_6,T_7,T_8)_1]}{1TS}\right)^1 \rightarrow \left(\begin{matrix}\frac{T_{28a}[(M_2, M_3)_1 \text{ and } (T_6,T_7,T_8)_1]}{2TS}\\ \frac{T_{28b}[(M_4)_1 \text{ and } (T_6,T_7,T_8)_1]}{1TS}\end{matrix}\right)^1 \rightarrow \left(\frac{T_{210a}[(M_2, M_3,M_4)_1 \text{ and } (T_6,T_7,T_8)_1]}{2TS}\right)^1 \rightarrow$

$\left(\frac{T_{211a}[(M_1,M_2, M_3,M_4)_1 \text{ and } (T_1)_1]}{2TS}\right)^1 \rightarrow \left(\frac{T_{212a}[(M_1,M_2, M_3,M_4)_1 \text{ and } (T_1)_1]}{2TS}\right)^1 \rightarrow \left(\begin{matrix}\frac{T_{213a}[(M_2, M_3)_1 \text{ and } (T_{12})_1]}{2TS}\\ \frac{T_{213b}[(M_4)_1 \text{ and } (T_{12})_1]}{1TS}\end{matrix}\right)^1 \rightarrow$

$\left(\begin{matrix}\frac{T_{214a}[(M_2, M_3)_1 \text{ and } (T_6,T_7,T_8)_1]}{2TS}\\ \frac{T_{214b}[(M_4)_1 \text{ and } (T_6,T_7,T_8)_1]}{1TS}\end{matrix}\right)^1 \rightarrow \left(\frac{T_{215a}[(M_1,M_2, M_3,M_4)_1 \text{ and } (T_1)_1]}{1TS}\right)^1 \rightarrow \left(\frac{T_{216a}[(M_1,M_2, M_3,M_4)_1 \text{ and } (T_1)_1]}{1TS}\right)^1$

Job #3: $\left(\frac{T_{31a}[(M_1,M_2, M_3,M_4)_1 \text{ and } (T_2)_1]}{1TS}\right)^1 \rightarrow \left(\frac{T_{32a}[(M_2, M_3,M_4)_1 \text{ and } (T_6,T_7,T_8)_1]}{1TS}\right)^1 \rightarrow \left(\begin{matrix}\frac{T_{33a}[(M_2,M_4)_1 \text{ and } (T_6,T_7,T_8)_1]}{2TS}\\ \frac{T_{22b}[(M_3)_1 \text{ and } (T_6,T_7,T_8)_1]}{1TS}\end{matrix}\right)^1 \rightarrow$

$\left(\begin{matrix}\frac{T_{34a}[(M_2,M_4)_1 \text{ and } (T_6,T_7,T_8)_1]}{2TS}\\ \frac{T_{34b}[(M_3)_1 \text{ and } (T_6,T_7,T_8)_1]}{1TS}\end{matrix}\right)^1 \rightarrow \left(\frac{T_{35a}[(M_1,M_2, M_3,M_4)_1 \text{ and } (T_2,T_3,T_4)_1]}{2TS}\right)^1 \rightarrow \left(\begin{matrix}\frac{T_{36a}[(M_2,M_3)_1 \text{ and } (T_9)_1]}{2TS}\\ \frac{T_{36b}[(M_4)_1 \text{ and } (T_9)_1]}{1TS}\end{matrix}\right)^1 \rightarrow$

$\left(\begin{matrix}\frac{T_{37a}[(M_2,M_3,M_5)_1 \text{ and } (T_{10})_1]}{2TS}\\ \frac{T_{37b}[(M_4)_1 \text{ and } (T_{10})_1]}{1TS}\end{matrix}\right)^1 \rightarrow \left(\frac{T_{38a}[(M_2, M_3,M_4)_1 \text{ and } (T_6,T_7,T_8)_1]}{1TS}\right)^1 \rightarrow \left(\frac{T_{39a}[(M_2, M_3,M_4)_1 \text{ and } (T_7,T_8,T_{11})_1]}{1TS}\right)^1 \rightarrow$

$\left(\frac{T_{310a}[(M_2, M_3,M_4)_1 \text{ and } (T_6,T_7,T_8,T_{11})_1]}{2TS}\right)^1 \rightarrow \left(\frac{T_{311a}[(M_2, M_3,M_4)_1 \text{ and } (T_7,T_8)_1]}{2TS}\right)^1 \rightarrow \left(\frac{T_{312a}[(M_1,M_2, M_3,M_4)_1 \text{ and } (T_1)_1]}{1TS}\right)^1 \rightarrow$

$\left(\frac{T_{313a}[(M_2, M_3,M_4)_1 \text{ and } (T_5)_1]}{1TS}\right)^1 \rightarrow \left(\frac{T_{314a}[(M_1,M_2, M_3,M_4)_1 \text{ and } (T_9)_1]}{1TS}\right)^1$

Job #4: $\left(\frac{T_{41a}[(M_2,M_4)_1 \text{ and } (T_9)_1]}{2TS}\right)^1 \rightarrow \left(\frac{T_{42a}[(M_1, M_2,M_4)_1 \text{ and } (T_2)_1]}{2TS}\right)^1 \rightarrow \left(\frac{T_{43a}[(M_2,M_4)_1 \text{ and } (T_6,T_9)_1]}{1TS}\right)^1 \rightarrow$

$\left(\frac{T_{44a}[(M_2,M_4)_1 \text{ and } (T_1,T_9)_1]}{2TS}\right)^1 \rightarrow \left(\frac{T_{45a}[(M_2,M_4)_1 \text{ and } (T_9,T_{10})_1]}{1TS}\right)^1 \rightarrow \left(\frac{T_{46a}[(M_2,M_4)_1 \text{ and } (T_1,T_9)_1]}{1TS}\right)^1 \rightarrow$

$\left(\frac{T_{47a}[(M_1,M_2,M_4)_1 \text{ and } (T_2)_1]}{1TS}\right)^1$

XIII. Testing 13

Job #1: $\left(\begin{matrix}\frac{T_{11a}[(M_2, M_3)_1 \text{ and } (T_6,T_7,T_8)_1]}{3TS}\\ \frac{T_{11b}[(M_4)_1 \text{ and } (T_6,T_7,T_8)_1]}{2TS}\end{matrix}\right)^1 \rightarrow \left(\begin{matrix}\frac{T_{12a}[(M_2, M_3)_1 \text{ and } (T_6,T_7,T_8)_1]}{3TS}\\ \frac{T_{12b}[(M_4)_1 \text{ and } (T_6,T_7,T_8)_1]}{2TS}\end{matrix}\right)^1 \rightarrow \left(\begin{matrix}\frac{T_{13a}[(M_2, M_3)_1 \text{ and } (T_6,T_7,T_8)_1]}{2TS}\\ \frac{T_{13b}[(M_4)_1 \text{ and } (T_6,T_7,T_8)_1]}{1TS}\end{matrix}\right)^1 \rightarrow$

$$\begin{pmatrix} \frac{T_{14a}[(M_2, M_3)_1 \text{ and } (T_6,T_7)_1]}{3TS} \\ \frac{T_{14b}[(M_4)_1 \text{ and } (T_6,T_7)_1]}{2TS} \end{pmatrix}^1 \rightarrow \left(\frac{T_{15a}[(M_2, M_3,M_4)_1 \text{ and } (T_7,T_8)_1]}{1TS}\right)^1 \rightarrow \left(\frac{T_{16a}[(M_2, M_3,M_4)_1 \text{ and } (T_7,T_8)_1]}{1TS}\right)^1 \rightarrow$$

$$\begin{pmatrix} \frac{T_{17a}[(M_2, M_3)_1 \text{ and } (T_6,T_7)_1]}{3TS} \\ \frac{T_{17b}[(M_4)_1 \text{ and } (T_6,T_7)_1]}{2TS} \end{pmatrix}^1 \rightarrow \left(\frac{T_{18a}[(M_1,M_2, M_3,M_4)_1 \text{ and } (T_2)_1]}{1TS}\right)^1 \rightarrow \left(\frac{T_{19a}[(M_2, M_3,M_4)_1 \text{ and } (T_9)_1]}{2TS}\right)^1 \rightarrow$$

$$\begin{pmatrix} \frac{T_{110a}[(M_1,M_2, M_3)_1 \text{ and } (T_2,T_3,T_4)_1]}{3TS} \\ \frac{T_{110b}[(M_4)_1 \text{ and } (T_2,T_3,T_4)_1]}{2TS} \end{pmatrix}^1 \rightarrow \left(\frac{T_{111a}[(M_2, M_3,M_4)_1 \text{ and } (T_9)_1]}{2TS}\right)^1 \rightarrow \left(\frac{T_{112a}[(M_2,M_3, M_4,M_5)_1 \text{ and } (T_{10})_1]}{2TS}\right)^1 \rightarrow$$

$$\left(\frac{T_{113a}[(M_1,M_2, M_3,M_4)_1 \text{ and } (T_1)_1]}{2TS}\right)^1 \rightarrow \begin{pmatrix} \frac{T_{114a}[(M_2, M_3)_1 \text{ and } (T_5)_1]}{2TS} \\ \frac{T_{114b}[(M_4)_1 \text{ and } (T_5)_1]}{1TS} \end{pmatrix}^1 \rightarrow \begin{pmatrix} \frac{T_{115a}[(M_2, M_3)_1 \text{ and } (T_7,T_8)_1]}{2TS} \\ \frac{T_{115b}[(M_4)_1 \text{ and } (T_7,T_8)_1]}{1TS} \end{pmatrix}^1 \rightarrow$$

$$\begin{pmatrix} \frac{T_{116a}[(M_1,M_2, M_3)_1 \text{ and } (T_2,T_3,T_4)_1]}{2TS} \\ \frac{T_{116b}[(M_4)_1 \text{ and } (T_2,T_3,T_4)_1]}{1TS} \end{pmatrix}^1 \rightarrow \left(\frac{T_{117a}[(M_2,M_3, M_4)_1 \text{ and } (T_9)_1]}{1TS}\right)^1 \rightarrow \left(\frac{T_{118a}[(M_2,M_3, M_4,M_5)_1 \text{ and } (T_{10})_1]}{1TS}\right)^1 \rightarrow$$

$$\left(\frac{T_{119a}[(M_2,M_3, M_4)_1 \text{ and } (T_9)_1]}{1TS}\right)^1 \rightarrow \left(\frac{T_{112a}[(M_2,M_3, M_4,M_5)_1 \text{ and } (T_{10})_1]}{1TS}\right)^1$$

Job #2: $\left(\frac{T_{21a}[(M_2, M_3,M_4)_1 \text{ and } (T_6,T_7,T_8)_1]}{1TS}\right)^1 \rightarrow \begin{pmatrix} \frac{T_{22a}[(M_1)_1 \text{ and } (T_6,T_7,T_8)_1]}{3TS} \\ \frac{T_{22b}[(M_2, M_3,M_4)_1 \text{ and } (T_6,T_7,T_8)_1]}{2TS} \end{pmatrix}^1 \rightarrow \left(\frac{T_{23a}[(M_2, M_3,M_4)_1 \text{ and } (T_1)_1]}{1TS}\right)^1 \rightarrow$

$$\begin{pmatrix} \frac{T_{24a}[(M_2, M_3)_1 \text{ and } (T_5,T_6,T_{11})_1]}{2TS} \\ \frac{T_{24b}[(M_4)_1 \text{ and } (T_5,T_6,T_{11})_1]}{1TS} \end{pmatrix}^1 \rightarrow \left(\frac{T_{25a}[(M_1,M_2, M_3,M_4)_1 \text{ and } (T_2)_1]}{2TS}\right)^1 \rightarrow \left(\frac{T_{26a}[(M_2, M_3,M_4)_1 \text{ and } (T_6,T_7,T_8)_1]}{1TS}\right)^1 \rightarrow$$

$$\left(\frac{T_{27a}[(M_2, M_3,M_4)_1 \text{ and } (T_6,T_7,T_8)_1]}{1TS}\right)^1 \rightarrow \begin{pmatrix} \frac{T_{28a}[(M_2, M_3)_1 \text{ and } (T_6,T_7,T_8)_1]}{2TS} \\ \frac{T_{28b}[(M_4)_1 \text{ and } (T_6,T_7,T_8)_1]}{1TS} \end{pmatrix}^1 \rightarrow \left(\frac{T_{210a}[(M_2, M_3,M_4)_1 \text{ and } (T_6,T_7,T_8)_1]}{2TS}\right)^1 \rightarrow$$

$$\left(\frac{T_{211a}[(M_1,M_2, M_3,M_4)_1 \text{ and } (T_1)_1]}{2TS}\right)^1 \rightarrow \left(\frac{T_{212a}[(M_1,M_2, M_3,M_4)_1 \text{ and } (T_1)_1]}{2TS}\right)^1 \rightarrow \begin{pmatrix} \frac{T_{213a}[(M_2, M_3)_1 \text{ and } (T_{12})_1]}{2TS} \\ \frac{T_{213b}[(M_4)_1 \text{ and } (T_{12})_1]}{1TS} \end{pmatrix}^1 \rightarrow$$

$$\begin{pmatrix} \frac{T_{214a}[(M_2, M_3)_1 \text{ and } (T_6,T_7,T_8)_1]}{2TS} \\ \frac{T_{214b}[(M_4)_1 \text{ and } (T_6,T_7,T_8)_1]}{1TS} \end{pmatrix}^1 \rightarrow \left(\frac{T_{215a}[(M_1,M_2, M_3,M_4)_1 \text{ and } (T_1)_1]}{1TS}\right)^1 \rightarrow \left(\frac{T_{216a}[(M_1,M_2, M_3,M_4)_1 \text{ and } (T_1)_1]}{1TS}\right)^1$$

Job #3: $\left(\frac{T_{31a}[(M_1,M_2, M_3,M_4)_1 \text{ and } (T_2)_1]}{1TS}\right)^1 \rightarrow \left(\frac{T_{32a}[(M_2, M_3,M_4)_1 \text{ and } (T_6,T_7,T_8)_1]}{1TS}\right)^1 \rightarrow \begin{pmatrix} \frac{T_{33a}[(M_2,M_4)_1 \text{ and } (T_6,T_7,T_8)_1]}{2TS} \\ \frac{T_{22b}[(M_3)_1 \text{ and } (T_6,T_7,T_8)_1]}{1TS} \end{pmatrix}^1 \rightarrow$

$$\begin{pmatrix} \frac{T_{34a}[(M_2,M_4)_1 \text{ and } (T_6,T_7,T_8)_1]}{2TS} \\ \frac{T_{34b}[(M_3)_1 \text{ and } (T_6,T_7,T_8)_1]}{1TS} \end{pmatrix}^1 \rightarrow \left(\frac{T_{35a}[(M_1,M_2, M_3,M_4)_1 \text{ and } (T_2,T_3,T_4)_1]}{2TS}\right)^1 \rightarrow \begin{pmatrix} \frac{T_{36a}[(M_2,M_3)_1 \text{ and } (T_9)_1]}{2TS} \\ \frac{T_{36b}[(M_4)_1 \text{ and } (T_9)_1]}{1TS} \end{pmatrix}^1 \rightarrow$$

$$\begin{pmatrix} \frac{T_{37a}[(M_2,M_3,M_5)_1 \text{ and } (T_{10})_1]}{2TS} \\ \frac{T_{37b}[(M_4)_1 \text{ and } (T_{10})_1]}{1TS} \end{pmatrix}^1 \rightarrow \left(\frac{T_{38a}[(M_2, M_3,M_4)_1 \text{ and } (T_6,T_7,T_8)_1]}{1TS}\right)^1 \rightarrow \left(\frac{T_{39a}[(M_2, M_3,M_4)_1 \text{ and } (T_7,T_8,T_{11})_1]}{1TS}\right)^{11} \rightarrow$$

$\left(\frac{T_{310a}[(M_2, M_3,M_4)_1 \text{ and } (T_6,T_7,T_8,T_{11})_1]}{2TS}\right)^1 \rightarrow \left(\frac{T_{311a}[(M_2, M_3,M_4)_1 \text{ and } (T_7,T_8)_1]}{2TS}\right)^1 \rightarrow \left(\frac{T_{312a}[(M_1,M_2, M_3,M_4)_1 \text{ and } (T_1)_1]}{1TS}\right)^1 \rightarrow$
$\left(\frac{T_{313a}[(M_2, M_3,M_4)_1 \text{ and } (T_5)_1]}{1TS}\right)^1 \rightarrow \left(\frac{T_{314a}[(M_1,M_2, M_3,M_4)_1 \text{ and } (T_9)_1]}{1TS}\right)^1$

Job #4: $\left(\frac{T_{41a}[(M_2,M_4)_1 \text{ and } (T_9)_1]}{2TS}\right)^1 \rightarrow \left(\frac{T_{42a}[(M_1, M_2,M_4)_1 \text{ and } (T_2)_1]}{2TS}\right)^1 \rightarrow \left(\frac{T_{43a}[(M_2,M_4)_1 \text{ and } (T_6,T_9)_1]}{1TS}\right)^1 \rightarrow$
$\left(\frac{T_{44a}[(M_2,M_4)_1 \text{ and } (T_1,T_9)_1]}{2TS}\right)^1 \rightarrow \left(\frac{T_{45a}[(M_2,M_4)_1 \text{ and } (T_9,T_{10})_1]}{1TS}\right)^1 \rightarrow \left(\frac{T_{46a}[(M_2,M_4)_1 \text{ and } (T_1,T_9)_1]}{1TS}\right)^1 \rightarrow$
$\left(\frac{T_{47a}[(M_1,M_2,M_4)_1 \text{ and } (T_2)_1]}{1TS}\right)^1$

XIV．Testing 14

Job #1: $\left(\begin{matrix}\frac{T_{11a}[(M_2, M_3)_1]}{3TS}\\ \frac{T_{11b}[(M_4)_1]}{2TS}\end{matrix}\right)^1 \rightarrow \left(\begin{matrix}\frac{T_{12a}[(M_2, M_3)_1]}{3TS}\\ \frac{T_{12b}[(M_4)_1]}{2TS}\end{matrix}\right)^1 \rightarrow \left(\begin{matrix}\frac{T_{13a}[(M_2, M_3)_1]}{2TS}\\ \frac{T_{13b}[(M_4)_1]}{1TS}\end{matrix}\right)^1 \rightarrow \left(\begin{matrix}\frac{T_{14a}[(M_2, M_3)_1]}{3TS}\\ \frac{T_{14b}[(M_4)_1]}{2TS}\end{matrix}\right)^1 \rightarrow \left(\frac{T_{15a}[(M_2, M_3,M_4)_1]}{1TS}\right)^1 \rightarrow$

$\left(\frac{T_{16a}[(M_2, M_3,M_4)_1]}{1TS}\right)^1 \rightarrow \left(\begin{matrix}\frac{T_{17a}[(M_2, M_3)_1]}{3TS}\\ \frac{T_{17b}[(M_4)_1]}{2TS}\end{matrix}\right)^1 \rightarrow \left(\frac{T_{18a}[(M_1,M_2, M_3,M_4)_1]}{1TS}\right)^1 \rightarrow \left(\frac{T_{19a}[(M_2, M_3,M_4)_1]}{2TS}\right)^1 \rightarrow$

$\left(\begin{matrix}\frac{T_{110a}[(M_1,M_2, M_3)_1]}{3TS}\\ \frac{T_{110b}[(M_4)_1]}{2TS}\end{matrix}\right)^1 \rightarrow \left(\frac{T_{111a}[(M_2, M_3,M_4)_1]}{2TS}\right)^1 \rightarrow \left(\frac{T_{112a}[(M_2,M_3, M_4,M_5)_1]}{2TS}\right)^1 \rightarrow \left(\frac{T_{113a}[(M_1,M_2, M_3,M_4)_1]}{2TS}\right)^1 \rightarrow$

$\left(\begin{matrix}\frac{T_{114a}[(M_2, M_3)_1]}{2TS}\\ \frac{T_{114b}[(M_4)_1]}{1TS}\end{matrix}\right)^1 \rightarrow \left(\begin{matrix}\frac{T_{115a}[(M_2, M_3)_1]}{2TS}\\ \frac{T_{115b}[(M_4)_1]}{1TS}\end{matrix}\right)^1 \rightarrow \left(\begin{matrix}\frac{T_{116a}[(M_1,M_2, M_3)_1]}{2TS}\\ \frac{T_{116b}[(M_4)_1]}{1TS}\end{matrix}\right)^1 \rightarrow \left(\frac{T_{117a}[(M_2,M_3, M_4)_1]}{1TS}\right)^1 \rightarrow$

$\left(\frac{T_{118a}[(M_2,M_3, M_4,M_5)_1]}{1TS}\right)^1 \rightarrow \left(\frac{T_{119a}[(M_2,M_3, M_4)_1]}{1TS}\right)^1 \rightarrow \left(\frac{T_{112a}[(M_2,M_3, M_4,M_5)_1]}{1TS}\right)^1$

Job #2: $\left(\frac{T_{21a}[(M_2, M_3,M_4)_1]}{1TS}\right)^1 \rightarrow \left(\begin{matrix}\frac{T_{22a}[(M_1)_1]}{3TS}\\ \frac{T_{22b}[(M_2, M_3,M_4)_1]}{2TS}\end{matrix}\right)^1 \rightarrow \left(\frac{T_{23a}[(M_2, M_3,M_4)_1]}{1TS}\right)^1 \rightarrow \left(\begin{matrix}\frac{T_{24a}[(M_2, M_3)_1]}{2TS}\\ \frac{T_{24b}[(M_4)_1]}{1TS}\end{matrix}\right)^1 \rightarrow$

$\left(\frac{T_{25a}[(M_1,M_2, M_3,M_4)_1]}{2TS}\right)^1 \rightarrow \left(\frac{T_{26a}[(M_2, M_3,M_4)_1]}{1TS}\right)^1 \rightarrow \left(\frac{T_{27a}[(M_2, M_3,M_4)_1]}{1TS}\right)^1 \rightarrow \left(\begin{matrix}\frac{T_{28a}[(M_2, M_3)_1]}{2TS}\\ \frac{T_{28b}[(M_4)_1]}{1TS}\end{matrix}\right)^1 \rightarrow$

$\left(\frac{T_{210a}[(M_2, M_3,M_4)_1]}{2TS}\right)^1 \rightarrow \left(\frac{T_{211a}[(M_1,M_2, M_3,M_4)_1]}{2TS}\right)^1 \rightarrow \left(\frac{T_{212a}[(M_1,M_2, M_3,M_4)_1]}{2TS}\right)^1 \rightarrow \left(\begin{matrix}\frac{T_{213a}[(M_2, M_3)_1]}{2TS}\\ \frac{T_{213b}[(M_4)_1]}{1TS}\end{matrix}\right)^1 \rightarrow$

$\left(\begin{matrix}\frac{T_{214a}[(M_2, M_3)_1]}{2TS}\\ \frac{T_{214b}[(M_4)_1\ ]}{1TS}\end{matrix}\right)^1 \rightarrow \left(\frac{T_{215a}[(M_1,M_2, M_3,M_4)_1]}{1TS}\right)^1 \rightarrow \left(\frac{T_{216a}[(M_1,M_2, M_3,M_4)_1]}{1TS}\right)^1$

Job #3: $\left(\frac{T_{31a}[(M_1,M_2, M_3,M_4)_1]}{1TS}\right)^1 \rightarrow \left(\frac{T_{32a}[(M_2, M_3,M_4)_1]}{1TS}\right)^1 \rightarrow \left(\begin{matrix}\frac{T_{33a}[(M_2,M_4)_1]}{2TS}\\ \frac{T_{22b}[(M_3)_1]}{1TS}\end{matrix}\right)^1 \rightarrow \left(\begin{matrix}\frac{T_{34a}[(M_2,M_4)_1]}{2TS}\\ \frac{T_{34b}[(M_3)_1]}{1TS}\end{matrix}\right)^1 \rightarrow$

$$\left(\frac{T_{35a}[(M_1,M_2,M_3,M_4)_1]}{2TS}\right)^1 \rightarrow \begin{pmatrix} \frac{T_{36a}[(M_2,M_3)_1]}{2TS} \\ \frac{T_{36b}[(M_4)_1]}{1TS} \end{pmatrix}^1 \rightarrow \begin{pmatrix} \frac{T_{37a}[(M_2,M_3,M_5)_1]}{2TS} \\ \frac{T_{37b}[(M_4)_1]}{1TS} \end{pmatrix}^1 \rightarrow \left(\frac{T_{38a}[(M_2,M_3,M_4)_1]}{1TS}\right)^1 \rightarrow \left(\frac{T_{39a}[(M_2,M_3,M_4)_1]}{1TS}\right)^1 \rightarrow$$

$$\left(\frac{T_{310a}[(M_2,M_3,M_4)_1]}{2TS}\right)^1 \rightarrow \left(\frac{T_{311a}[(M_2,M_3,M_4)_1]}{2TS}\right)^1 \rightarrow \left(\frac{T_{312a}[(M_1,M_2,M_3,M_4)_1]}{1TS}\right)^1 \rightarrow \left(\frac{T_{313a}[(M_2,M_3,M_4)_1]}{1TS}\right)^1 \rightarrow$$

$$\left(\frac{T_{314a}[(M_1,M_2,M_3,M_4)_1]}{1TS}\right)^1$$

Job #4: $\left(\frac{T_{41a}[(M_2,M_4)_1]}{2TS}\right)^1 \rightarrow \left(\frac{T_{42a}[(M_1,M_2,M_4)_1]}{2TS}\right)^1 \rightarrow \left(\frac{T_{43a}[(M_2,M_4)_1]}{1TS}\right)^1 \rightarrow \left(\frac{T_{44a}[(M_2,M_4)_1]}{2TS}\right)^1 \rightarrow \left(\frac{T_{45a}[(M_2,M_4)_1]}{1TS}\right)^1 \rightarrow \left(\frac{T_{46a}[(M_2,M_4)_1]}{1TS}\right)^1 \rightarrow \left(\frac{T_{47a}[(M_1,M_2,M_4)_1]}{1TS}\right)^1$

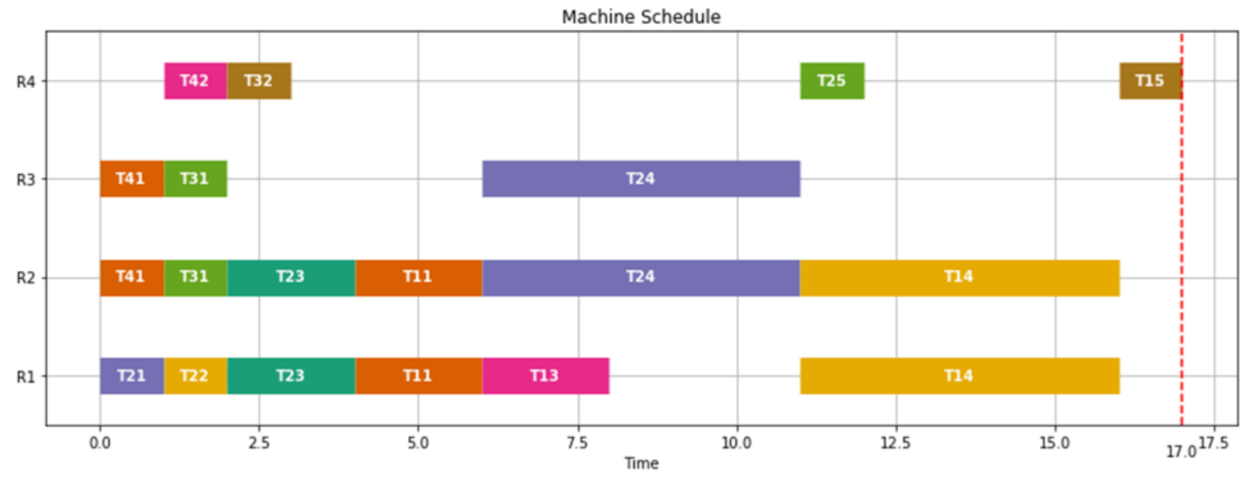
XV. Testing 15

Job #1: $\frac{T_{11}[(R_1,R_2,R_3)_2]}{2TS} \rightarrow \frac{T_{12}[(R_1,R_2,R_3)_1]}{2TS} \rightarrow \frac{T_{13}[(R_1,R_2,R_3)_2]}{5TS} \rightarrow \frac{T_{14}[(R_4)_1]}{1TS}$

Job #2: $\frac{T_{21}[(R_1,R_2,R_3)_1]}{1TS} \rightarrow \frac{T_{22}[(R_1,R_2)_1]}{1TS} \rightarrow \frac{T_{23}[(R_1)_1\text{and}(R_2,R_3)_1]}{2TS} \rightarrow \frac{T_{24}[(R_1,R_2,R_3)_2]}{5TS} \rightarrow \frac{T_{25}[(R_4)_1]}{1TS}$

Job #3: $\frac{T_{31}[(R_1,R_2,R_3)_2]}{1TS} \rightarrow \frac{T_{32}[(R_4)_1]}{1TS}$

Job #4: $\frac{T_{41}[(R_3)_1\text{and}(R_1,R_2)_1]}{1TS} \rightarrow \frac{T_{42}[(R_4)_1]}{1TS}$



XVI. Testing 16

Job #1: $\frac{T_{11}[(R_1,R_2,R_3)_2]}{2TS} \rightarrow \frac{T_{12}[(R_1,R_2,R_3)_2]}{1TS} \rightarrow \frac{T_{13}[(R_1,R_2,R_3)_1]}{2TS} \rightarrow \frac{T_{14}[(R_4)_1]}{1TS}$

Job #2: $\frac{T_{21}[(R_1,R_2,R_3)_1]}{1TS} \rightarrow \frac{T_{22}[(R_1)_1\text{and}(R_2,R_3)_1]}{2TS} \rightarrow \frac{T_{23}[(R_4)_1]}{1TS}$

Job #3: $\frac{T_{31}[(R_1,R_2)_1]}{1TS} \rightarrow \begin{pmatrix} \frac{T_{32a}[(R_1,R_2)_2]}{1TS} \\ \frac{T_{32b}[(R_3)_1]}{2TS} \end{pmatrix}^1 \rightarrow \frac{T_{33}[(R_3,R_4)_1]}{3TS}$

Job #4: $\frac{T_{41}[(R_1,R_2,R_3)_3]}{1TS} \rightarrow \frac{T_{42}[(R_1,R_2,R_3)_2]}{2TS}$

Job #5: $\frac{T_{51}[(R_1,R_2,R_3)_3]}{2TS} \rightarrow \frac{T_{52}[(R_1,R_2,R_3,R_5)_2]}{1TS}$

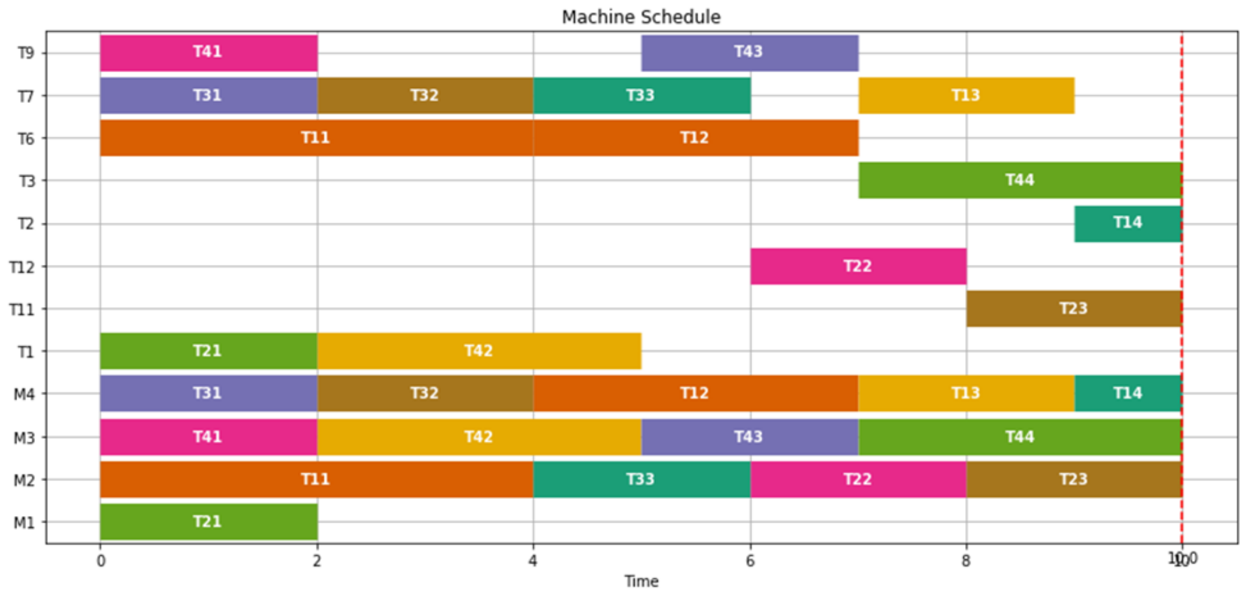
XVII. Testing 17

Job #1: $\left(\begin{array}{c}\frac{T_{1,1a}[(M_2,M_3)_1 \text{ and } (T_6,T_7)_1]}{4TS} \\ \frac{T_{1,1b}[(M_4)_1 \text{ and } (T_6,T_7)_1]}{3TS}\end{array}\right)^1 \rightarrow \left(\begin{array}{c}\frac{T_{1,2a}[(M_2,M_3)_1 \text{ and } (T_6,T_7)_1]}{4TS} \\ \frac{T_{1,2b}[(M_4)_1 \text{ and } (T_6,T_7)_1]}{3TS}\end{array}\right)^1 \rightarrow \left(\frac{T_{1,3a}[(M_2,M_3,M_4)_1 \text{ and } (T_6,T_7)_1]}{2TS}\right)^1 \rightarrow \left(\frac{T_{1,4a}[(M_4)_1 \text{ and } (T_2)_1]}{1TS}\right)^1$

Job #2: $\left(\begin{array}{c}\frac{T_{2,1b}[(M_1)_1 \text{ and } (T_1)_1]}{2TS} \\ \frac{T_{2,1a}[(M_2,M_3,M_4)_1 \text{ and } (T_1)_1]}{1TS}\end{array}\right)^1 \rightarrow \left(\frac{T_{2,2a}[(M_2,M_3,M_4)_1 \text{ and } (T_{12})_1]}{2TS}\right)^1 \rightarrow \left(\frac{T_{2,3a}[(M_2,M_3,M_4)_1 \text{ and } (T_{11})_1]}{2TS}\right)^1$

Job #3: $\left(\frac{T_{3,1a}[(M_2,M_3,M_4)_1 \text{ and } (T_7)_1]}{2TS}\right)^1 \rightarrow \left(\frac{T_{3,2a}[(M_2,M_3,M_4)_1 \text{ and } (T_7)_1]}{2TS}\right)^1 \rightarrow \left(\frac{T_{3,3a}[(M_2,M_3,M_4)_1 \text{ and } (T_7)_1]}{2TS}\right)^1$

Job #4: $\left(\begin{array}{c}\frac{T_{4,1a}[(M_2)_1 \text{ and } (T_9)_1]}{3TS} \\ \frac{T_{4,1b}[(M_3)_1 \text{ and } (T_9)_1]}{2TS}\end{array}\right)^1 \rightarrow \left(\frac{T_{4,2a}[(M_2,M_3)_1 \text{ and } (T_1,T_3)_1]}{3TS}\right)^1 \rightarrow \left(\frac{T_{4,3a}[(M_2,M_3)_1 \text{ and } (T_6,T_9)_1]}{2TS}\right)^1 \rightarrow \left(\begin{array}{c}\frac{T_{4,4a}[(M_2)_1 \text{ and } (T_3)_1]}{2TS} \\ \frac{T_{4,4b}[(M_3)_1 \text{ and } (T_3)_1]}{3TS}\end{array}\right)^1$



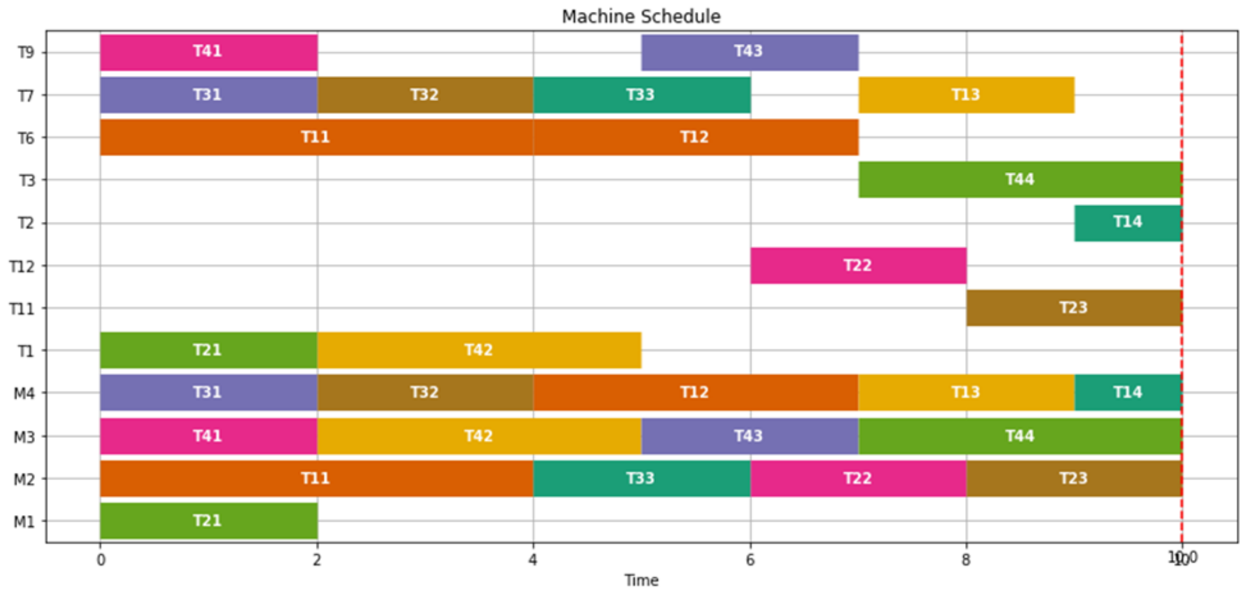
XVIII. Testing 18

Job #1: $\begin{pmatrix} \frac{T_{1,1a}[(M_2, M_3)_1 \text{ and } (T_6,T_7)_1]}{4TS} \\ \frac{T_{1,1b}[(M_4)_1 \text{ and } (T_6,T_7)_1]}{3TS} \end{pmatrix}^1 \rightarrow \begin{pmatrix} \frac{T_{1,2a}[(M_2, M_3)_1 \text{ and } (T_6,T_7)_1]}{4TS} \\ \frac{T_{1,2b}[(M_4)_1 \text{ and } (T_6,T_7)_1]}{3TS} \end{pmatrix}^1 \rightarrow \left(\frac{T_{1,3a}[(M_2, M_3,M_4)_1 \text{ and } (T_6,T_7)_1]}{2TS}\right)^1 \rightarrow \left(\frac{T_{1,4a}[(M_4)_1 \text{ and } (T_2)_1]}{1TS}\right)^1$

Job #2: $\begin{pmatrix} \frac{T_{2,1b}[(M_1)_1 \text{ and } (T_1)_1]}{2TS} \\ \frac{T_{2,1a}[(M_2, M_3,M_4)_1 \text{ and } (T_1)_1]}{1TS} \end{pmatrix}^1 \rightarrow \left(\frac{T_{2,2a}[(M_2, M_3,M_4)_1 \text{ and } (T_{12})_1]}{2TS}\right)^1 \rightarrow \left(\frac{T_{2,3a}[(M_2, M_3,M_4)_1 \text{ and } (T_6,T_7,T_{11})_1]}{2TS}\right)^1$

Job #3: $\left(\frac{T_{3,1a}[(M_2, M_3,M_4)_1 \text{ and } (T_7)_1]}{2TS}\right)^1 \rightarrow \left(\frac{T_{3,2a}[(M_2, M_3,M_4)_1 \text{ and } (T_7)_1]}{2TS}\right)^1 \rightarrow \left(\frac{T_{3,3a}[(M_2, M_3,M_4)_1 \text{ and } (T_7)_1]}{2TS}\right)^1$

Job #4: $\begin{pmatrix} \frac{T_{4,1a}[(M_2)_1 \text{ and } (T_9)_1]}{3TS} \\ \frac{T_{4,1b}[(M_3)_1 \text{ and } (T_9)_1]}{2TS} \end{pmatrix}^1 \rightarrow \left(\frac{T_{4,2a}[(M_2,M_3)_1 \text{ and } (T_1,T_3)_1]}{3TS}\right)^1 \rightarrow \left(\frac{T_{4,3a}[(M_2,M_3)_1 \text{ and } (T_6,T_9)_1]}{2TS}\right)^1 \rightarrow \begin{pmatrix} \frac{T_{4,4a}[(M_2)_1 \text{ and } (T_3)_1]}{2TS} \\ \frac{T_{4,4b}[(M_3)_1 \text{ and } (T_3)_1]}{3TS} \end{pmatrix}^1$



XIX. Testing 19

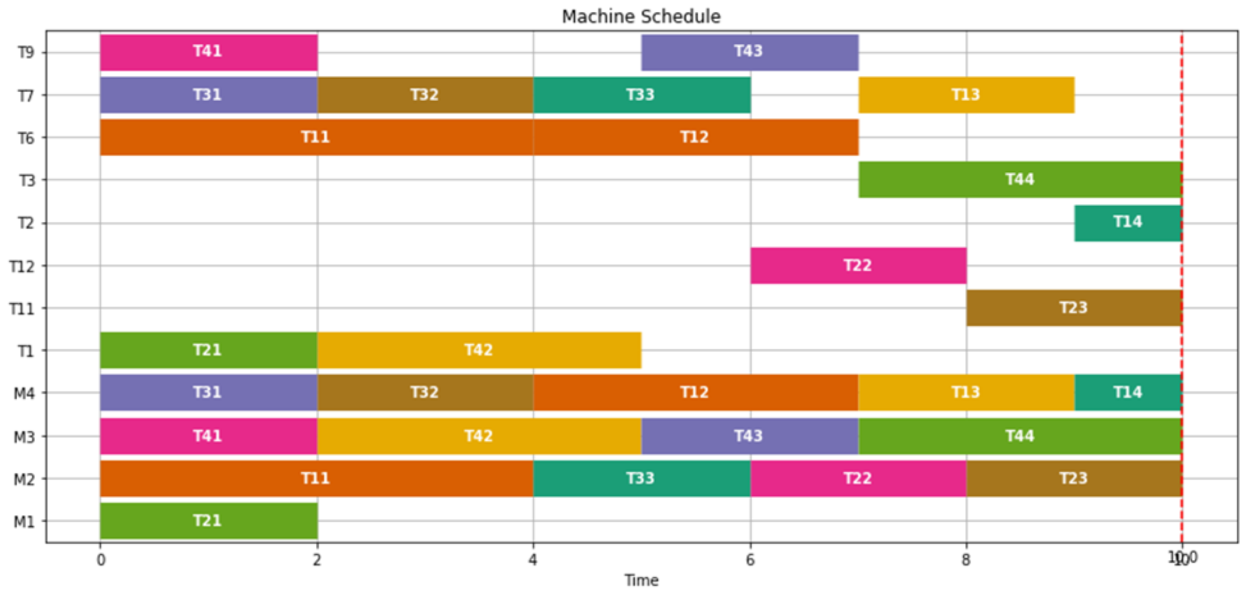
Job #1: $\begin{pmatrix} \frac{T_{1,1a}[(M_2)_1 \text{ and } (T_6,T_7)_1]}{4TS} \\ \frac{T_{1,1b}[(M_4)_1 \text{ and } (T_6,T_7)_1]}{3TS} \end{pmatrix}^1 \rightarrow \left(\frac{T_{1,2a}[(M_4)_1 \text{ and } (T_6,T_7)_1]}{2TS}\right)^1 \rightarrow \left(\frac{T_{1,3a}[(M_2, M_3,M_4)_1 \text{ and } (T_6,T_7)_1]}{2TS}\right)^1 \rightarrow \left(\frac{T_{1,4a}[(M_4)_1 \text{ and } (T_2)_1]}{1TS}\right)^1$

Job #2: $\left(\frac{T_{2,1a}[(M_2, M_3,M_4)_1 \text{ and } (T_1)_1]}{2TS}\right)^1 \rightarrow \left(\frac{T_{2,2a}[(M_2, M_3,M_4)_1 \text{ and } (T_{12})_1]}{2TS}\right)^1 \rightarrow \left(\frac{T_{2,3a}[(M_2)_1 \text{ and } (T_{11})_1]}{2TS}\right)^1$

Job #3: $\left(\frac{T_{3,1a}[(M_2, M_3,M_4)_1 \text{ and } (T_7)_1]}{2TS}\right)^1 \rightarrow \left(\frac{T_{3,2a}[(M_2, M_3,M_4)_1 \text{ and } (T_7)_1]}{2TS}\right)^1 \rightarrow \left(\frac{T_{3,3a}[(M_2, M_3,M_4)_1 \text{ and } (T_7)_1]}{2TS}\right)^1$

Job #4: $\left(\frac{T_{4,1a}[(M_3)_1 \text{ and } (T_9)_1]}{1TS}\right)^1 \rightarrow \left(\frac{T_{4,2a}[(M_2,M_3)_1 \text{ and } (T_1,T_3)_1]}{3TS}\right)^1 \rightarrow \left(\frac{T_{4,3a}[(M_2,M_3)_1 \text{ and } (T_6,T_9)_1]}{2TS}\right)^1 \rightarrow$

$$\left(\begin{matrix}\frac{T_{4,4a}[(M_2)_1 \text{ and } (T_3)_1]}{2TS} \\ \frac{T_{4,4b}[(M_3)_1 \text{ and } (T_3)_1]}{3TS}\end{matrix}\right)^1$$



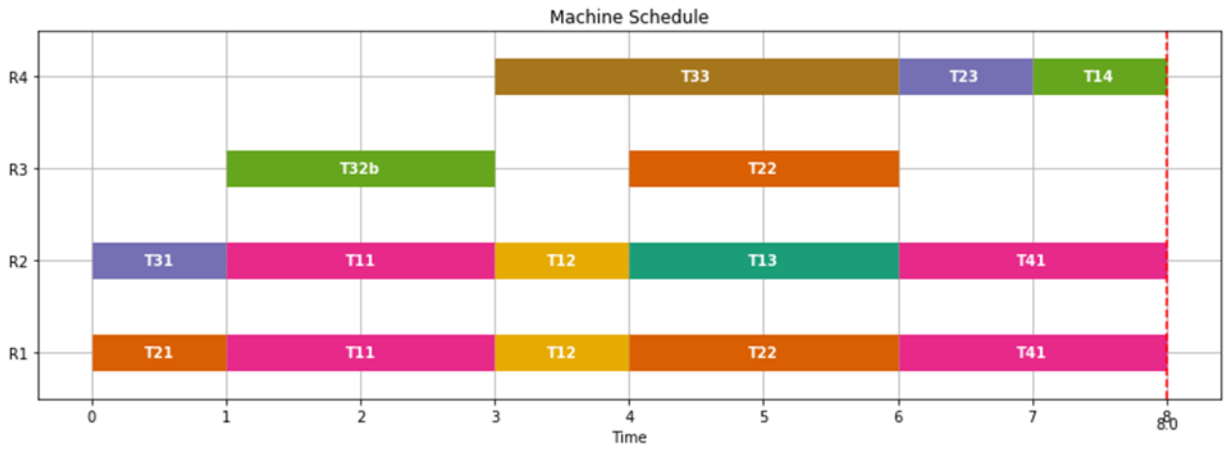
XX. Testing 20

Job #1: $\frac{T_{11}[(R_1,R_2,R_3)_2]}{2} \rightarrow \frac{T_{12}[(R_1,R_2,R_3)_2]}{1} \rightarrow \frac{T_{13}[(R_1,R_2,R_3)_1]}{2} \rightarrow \frac{T_{14}[(R_4)_1]}{1}$

Job #2: $\frac{T_{21}[(R_1,R_2,R_3)_1]}{1} \rightarrow \frac{T_{22}[(R_1)_1 \text{and} (R_2,R_3)_1]}{2} \rightarrow \frac{T_{23}[(R_4)_1]}{1}$

Job #3: $\frac{T_{31}[(R_1,R_2)_1]}{1} \rightarrow \left(\begin{matrix}\frac{T_{32a}[(R_1,R_2)_2]}{1} \\ \frac{T_{32b}[(R_3)_1]}{2}\end{matrix}\right)^1 \rightarrow \frac{T_{33}[(R_3,R_4)_1]}{3}$

Job #4: $\frac{T_{41}[(R_1,R_2,R_3)_2]}{2}$



XXI. Testing 21

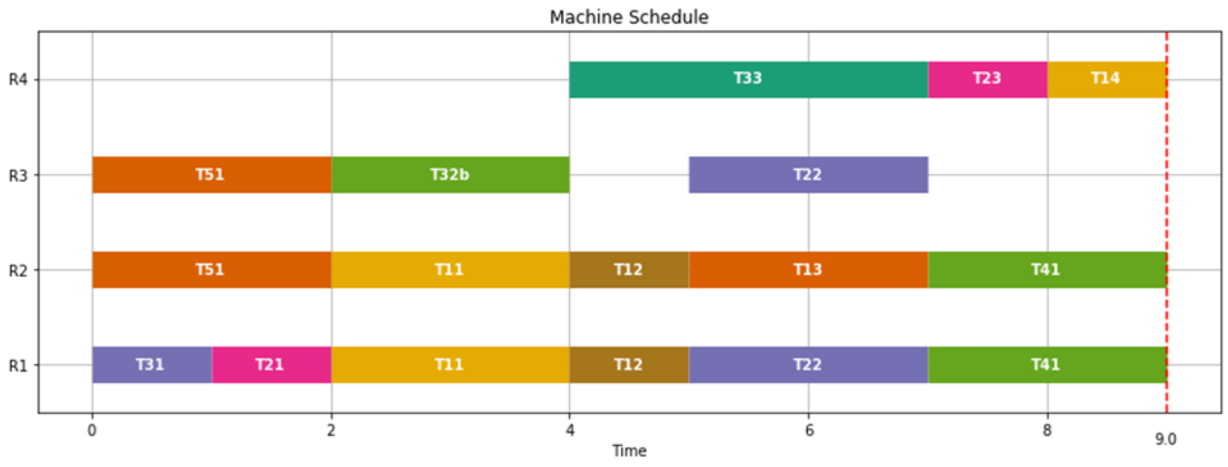
Job #1: $\frac{T_{11}[(R_1,R_2,R_3)_2]}{2} \rightarrow \frac{T_{12}[(R_1,R_2,R_3)_2]}{1} \rightarrow \frac{T_{13}[(R_1,R_2,R_3)_1]}{2} \rightarrow \frac{T_{14}[(R_4)_1]}{1}$

Job #2: $\frac{T_{21}[(R_1,R_2,R_3)_1]}{1} \rightarrow \frac{T_{22}[(R_1)_1 \text{and} (R_2,R_3)_1]}{2} \rightarrow \frac{T_{23}[(R_4)_1]}{1}$

Job #3: $\frac{T_{31}[(R_1,R_2)_1]}{1} \rightarrow \begin{pmatrix} \frac{T_{32a}[(R_1,R_2)_2]}{1} \\ \frac{T_{32b}[(R_3)_1]}{2} \end{pmatrix}^1 \rightarrow \frac{T_{33}[(R_3,R_4)_1]}{3}$

Job #4: $\frac{T_{41}[(R_1,R_2,R_3)_2]}{2}$

Job #5: $\frac{T_{51}[(R_1,R_2,R_3)_2]}{2}$

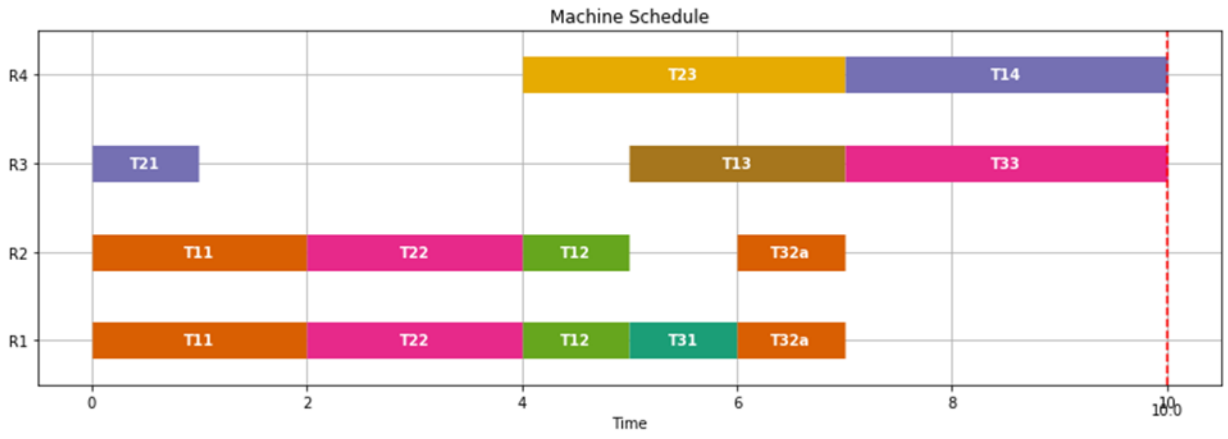


XXII. Testing 22

Job #1: $\frac{T_{11}[(R_1,R_2,R_3)_2]}{2} \rightarrow \frac{T_{12}[(R_1,R_2,R_3)_2]}{1} \rightarrow \frac{T_{13}[(R_1,R_2,R_3)_1]}{2} \rightarrow \frac{T_{14}[(R_4)_1]}{3}$

Job #2: $\frac{T_{21}[(R_1,R_2,R_3)_1]}{1} \rightarrow \frac{T_{22}[(R_1)_1 \text{and} (R_2,R_3)_1]}{2} \rightarrow \frac{T_{23}[(R_4)_1]}{3}$

Job #3: $\frac{T_{31}[(R_1,R_2)_1]}{1} \rightarrow \begin{pmatrix} \frac{T_{32a}[(R_1,R_2)_2]}{1} \\ \frac{T_{32b}[(R_3)_1]}{2} \end{pmatrix}^1 \rightarrow \frac{T_{33}[(R_3,R_4)_1]}{3}$



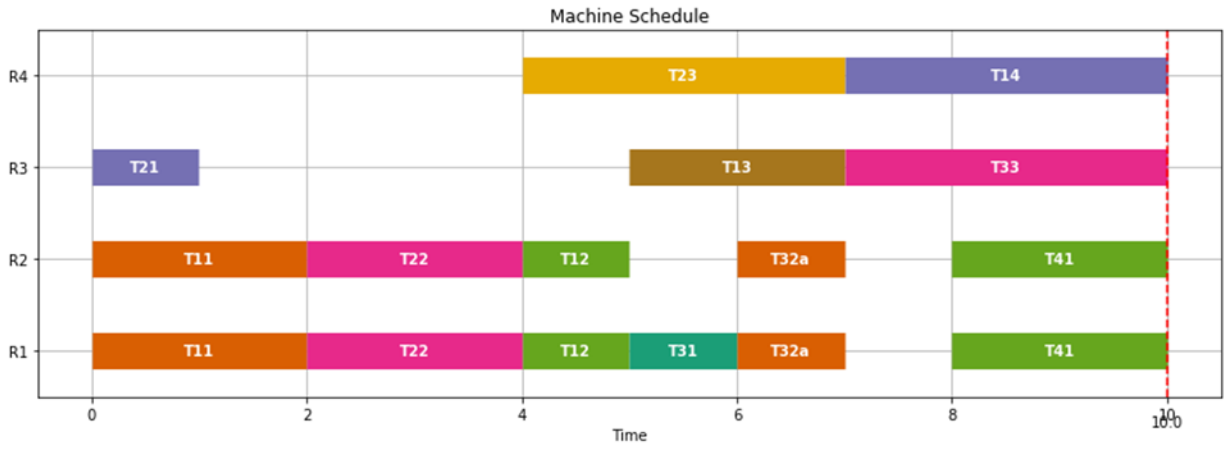
XXIII. Testing 23

Job #1: $\frac{T_{11}[(R_1,R_2,R_3)_2]}{2} \rightarrow \frac{T_{12}[(R_1,R_2,R_3)_2]}{1} \rightarrow \frac{T_{13}[(R_1,R_2,R_3)_1]}{2} \rightarrow \frac{T_{14}[(R_4)_1]}{3}$

Job #2: $\frac{T_{21}[(R_1,R_2,R_3)_1]}{1} \rightarrow \frac{T_{22}[(R_1)_1 \text{and} (R_2,R_3)_1]}{2} \rightarrow \frac{T_{23}[(R_4)_1]}{3}$

Job #3: $\frac{T_{31}[(R_1,R_2)_1]}{1} \rightarrow \begin{pmatrix} \frac{T_{32a}[(R_1,R_2)_2]}{1} \\ \frac{T_{32b}[(R_3)_1]}{2} \end{pmatrix}^1 \rightarrow \frac{T_{33}[(R_3,R_4)_1]}{3}$

Job #4: $\frac{T_{41}[(R_1,R_2,R_3)_2]}{2}$



XXIV. Testing 24

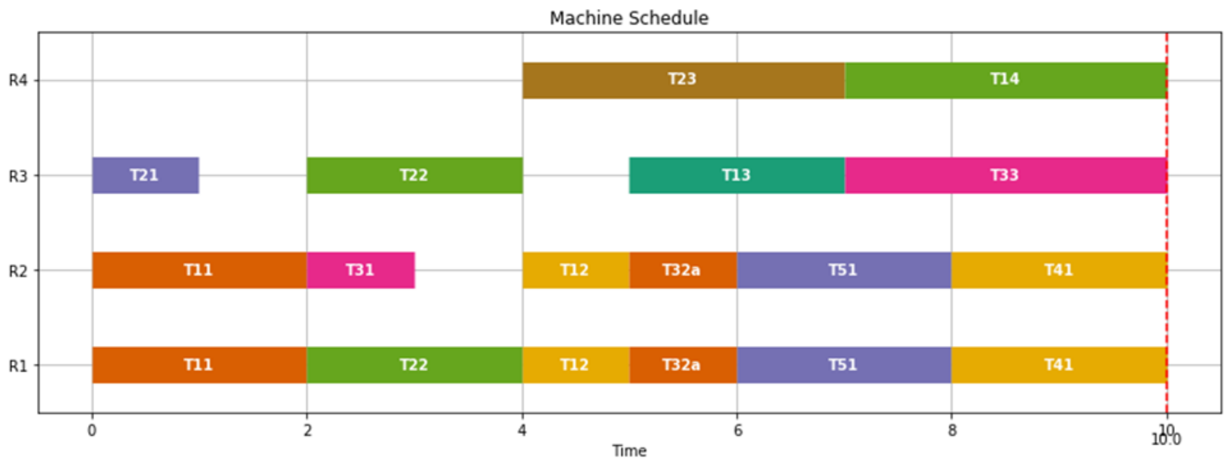
Job #1: $\frac{T_{11}[(R_1,R_2,R_3)_2]}{2} \rightarrow \frac{T_{12}[(R_1,R_2,R_3)_2]}{1} \rightarrow \frac{T_{13}[(R_1,R_2,R_3)_1]}{2} \rightarrow \frac{T_{14}[(R_4)_1]}{3}$

Job #2: $\frac{T_{21}[(R_1,R_2,R_3)_1]}{1} \rightarrow \frac{T_{22}[(R_1)_1 \text{and} (R_2,R_3)_1]}{2} \rightarrow \frac{T_{23}[(R_4)_1]}{3}$

Job #3: $\frac{T_{31}[(R_1,R_2)_1]}{1} \rightarrow \begin{pmatrix} \frac{T_{32a}[(R_1,R_2)_2]}{1} \\ \frac{T_{32b}[(R_3)_1]}{2} \end{pmatrix}^1 \rightarrow \frac{T_{33}[(R_3,R_4)_1]}{3}$

Job #4: $\frac{T_{41}[(R_1,R_2,R_3)_2]}{2}$

Job #5: $\frac{T_{51}[(R_1,R_2,R_3)_2]}{2}$



Appendix IV: The PPS Test Results Summary on Accuracy

Test Accuracy Summary (best among three length weight coefficients)					
Heuristics	Error Sum	Min Error	Max Error	Standard Deviation	Average Error
H2	76.79%	0.00%	14.29%	0.053831568	3.66%
H8	82.67%	0.00%	14.29%	0.053362191	3.94%
H16	96.14%	0.00%	14.29%	0.053670139	4.01%
H5	89.29%	0.00%	14.29%	0.056434276	4.25%
H19	106.14%	0.00%	20.00%	0.061793045	4.42%
H3	99.29%	0.00%	20.00%	0.065083392	4.73%
H28	116.46%	0.00%	20.00%	0.067476508	4.85%
H25	120.48%	0.00%	14.29%	0.056689463	5.02%
H15	128.05%	0.00%	14.29%	0.05304635	5.34%
H4	115.17%	0.00%	14.29%	0.055658769	5.48%
H18	140.37%	0.00%	18.18%	0.058554961	5.85%
H14	142.73%	0.00%	20.00%	0.057587423	5.95%
H17	145.96%	0.00%	14.29%	0.050042525	6.08%
H1	149.29%	0.00%	30.00%	0.093683853	7.11%
H13	172.19%	0.00%	30.00%	0.091927199	7.17%
H24	173.32%	0.00%	30.00%	0.082203899	7.22%
H12	178.46%	0.00%	30.00%	0.076173156	7.44%
H22	182.19%	0.00%	30.00%	0.090793192	7.59%
H6	167.31%	0.00%	28.57%	0.072708516	7.97%
H11	193.94%	0.00%	20.00%	0.072219182	8.08%
H9	190.72%	0.00%	28.57%	0.061908107	9.08%
H27	221.95%	0.00%	40.00%	0.108238797	9.25%
H10	210.56%	0.00%	28.57%	0.055782302	10.03%
H21	274.45%	0.00%	30.00%	0.097020496	11.44%
H7	342.98%	0.00%	40.00%	0.116260159	16.33%
H23	695.72%	5.56%	67.74%	0.17776048	28.99%
H20	711.26%	5.56%	70.97%	0.182550518	29.64%
H26	768.71%	5.56%	87.10%	0.226693545	32.03%

Test Accuracy Summary (different length weight coefficients)						
Heuristics	Length Weight	Error Sum	Min Error	Max Error	Standard Deviation	Average Error
H1	LW median	146.79%	0.00%	30.00%	0.116101709	7.73%
H1	LW high	66.79%	0.00%	14.29%	0.054035859	3.52%
H1	LW low	379.65%	0.00%	50.00%	0.215917236	19.98%
H2	LW median	116.79%	0.00%	20.00%	0.079669269	6.15%
H2	LW high	96.79%	0.00%	20.00%	0.073307066	5.09%

H2	LW low	425.39%	0.00%	60.00%	0.20627864	22.39%
H3	LW median	82.34%	0.00%	14.29%	0.063872671	4.33%
H3	LW high	96.79%	0.00%	20.00%	0.073307066	5.09%
H3	LW low	337.89%	0.00%	50.00%	0.199032284	17.78%
H4	LW median	132.67%	0.00%	30.00%	0.107201134	6.98%
H4	LW high	72.67%	0.00%	14.29%	0.05359337	3.82%
H4	LW low	377.15%	0.00%	60.00%	0.227042888	19.85%
H5	LW median	132.67%	0.00%	20.00%	0.090325798	6.98%
H5	LW high	92.67%	0.00%	20.00%	0.06424635	4.88%
H5	LW low	467.89%	0.00%	60.00%	0.211785288	24.63%
H6	LW median	239.09%	0.00%	28.57%	0.103530837	12.58%
H6	LW high	157.31%	0.00%	28.57%	0.074081227	8.28%
H6	LW low	503.60%	0.00%	60.00%	0.21305678	26.51%
H7	LW median	274.65%	0.00%	40.00%	0.117823431	14.46%
H7	LW high	264.65%	0.00%	40.00%	0.11744188	13.93%
H7	LW low	423.74%	0.00%	60.00%	0.213851589	22.30%
H8	LW median	72.67%	0.00%	14.29%	0.05359337	3.82%
H8	LW high	92.67%	0.00%	20.00%	0.06424635	4.88%
H8	LW low	266.55%	0.00%	50.00%	0.194635336	14.03%
H9	LW median	168.06%	0.00%	28.57%	0.066280593	8.85%
H9	LW high	157.31%	0.00%	28.57%	0.074081227	8.28%
H9	LW low	453.60%	0.00%	50.00%	0.188410498	23.87%
H10	LW median	238.06%	0.00%	28.57%	0.120058681	12.53%
H10	LW high	177.15%	0.00%	28.57%	0.069662802	9.32%
H10	LW low	487.89%	0.00%	60.00%	0.204165627	25.68%
H11	LW median	275.40%	0.00%	40.00%	0.132007015	14.49%
H11	LW high	201.21%	0.00%	36.36%	0.102077836	10.59%
H11	LW low	371.62%	0.00%	42.86%	0.166448838	19.56%
H12	LW median	219.18%	0.00%	40.00%	0.127558102	11.54%
H12	LW high	144.00%	0.00%	18.18%	0.06672139	7.58%
H12	LW low	393.70%	0.00%	42.86%	0.129630104	20.72%
H13	LW median	194.71%	0.00%	42.86%	0.137130947	10.25%
H13	LW high	89.69%	0.00%	14.29%	0.062333895	4.72%
H13	LW low	325.46%	0.00%	50.00%	0.146692817	17.13%
H14	LW median	209.92%	0.00%	30.00%	0.104197662	11.05%
H14	LW high	151.82%	0.00%	30.00%	0.089640345	7.99%
H14	LW low	319.95%	0.00%	40.00%	0.157068297	16.84%
H15	LW median	205.23%	0.00%	40.00%	0.112344169	10.80%
H15	LW high	114.64%	0.00%	18.18%	0.059177995	6.03%
H15	LW low	418.39%	0.00%	50.00%	0.126877852	22.02%
H16	LW median	146.14%	0.00%	20.00%	0.094973115	7.69%
H16	LW high	106.14%	0.00%	20.00%	0.06475161	5.59%
H16	LW low	423.45%	0.00%	50.00%	0.130585361	22.29%

H17	LW median	192.74%	0.00%	30.00%	0.090580368	10.14%
H17	LW high	147.55%	0.00%	18.18%	0.063333126	7.77%
H17	LW low	233.49%	0.00%	30.00%	0.114784161	12.29%
H18	LW median	161.10%	0.00%	40.00%	0.100517064	8.48%
H18	LW high	132.16%	0.00%	28.57%	0.076243832	6.96%
H18	LW low	340.30%	0.00%	40.00%	0.106109163	17.91%
H19	LW median	104.92%	0.00%	14.29%	0.06548711	5.52%
H19	LW high	125.82%	0.00%	20.00%	0.071838329	6.62%
H19	LW low	347.65%	0.00%	50.00%	0.18967755	18.30%
H20	LW median	534.04%	5.56%	70.97%	0.193136428	28.11%
H20	LW high	567.68%	5.56%	80.65%	0.214868683	29.88%
H20	LW low	553.39%	5.56%	80.65%	0.21783028	29.13%
H21	LW median	242.68%	0.00%	30.00%	0.11708745	12.77%
H21	LW high	184.85%	0.00%	20.00%	0.077665972	9.73%
H21	LW low	365.80%	0.00%	30.00%	0.092706741	19.25%
H22	LW median	277.62%	0.00%	50.00%	0.191346926	14.61%
H22	LW high	99.69%	0.00%	14.29%	0.062348484	5.25%
H22	LW low	375.14%	0.00%	50.00%	0.166974759	19.74%
H23	LW median	580.58%	5.56%	87.10%	0.232079338	30.56%
H23	LW high	580.58%	5.56%	87.10%	0.232079338	30.56%
H23	LW low	518.50%	5.56%	67.74%	0.186594926	27.29%
H24	LW median	201.95%	0.00%	22.58%	0.085814202	10.63%
H24	LW high	159.04%	0.00%	20.00%	0.067677392	8.37%
H24	LW low	353.30%	0.00%	30.00%	0.101758463	18.59%
H25	LW median	162.59%	0.00%	20.00%	0.094948821	8.56%
H25	LW high	142.27%	0.00%	20.00%	0.071557047	7.49%
H25	LW low	408.11%	0.00%	50.00%	0.144216269	21.48%
H26	LW median	580.58%	5.56%	87.10%	0.232079338	30.56%
H26	LW high	580.58%	5.56%	87.10%	0.232079338	30.56%
H26	LW low	571.49%	5.56%	87.10%	0.233725333	30.08%
H27	LW median	233.95%	0.00%	30.00%	0.113169757	12.31%
H27	LW high	219.46%	0.00%	28.57%	0.108478598	11.55%
H27	LW low	325.79%	0.00%	30.00%	0.105947041	17.15%
H28	LW median	128.47%	0.00%	14.29%	0.07665969	6.76%
H28	LW high	149.69%	0.00%	20.00%	0.103908679	7.88%
H28	LW low	360.55%	0.00%	50.00%	0.175904418	18.98%

Appendix V: The PPS Test Results

Test Instances		T1		
Job Number		2		
Operation Number		7		
Edge Number		111		
Node Number		24		
Total Length		10		
Average Length		1.428571429		
Optimum Time Slot		7		
LWs	Makespan	Computation Time (s)	Difference	Error Rate
IP	7	6.590471577	0	0.00%
H1 LW=1	7	0.171875	0	0.00%
H1 LW=10	7	0.171875	0	0.00%
H1 LW=0.001	8	0.234375	1	14.29%
H2 LW=1	7	0.140625	0	0.00%
H2 LW=10	7	0.140625	0	0.00%
H2 LW=0.001	8	0.15625	1	14.29%
H3 LW=1	7	0.140625	0	0.00%
H3 LW=10	7	0.15625	0	0.00%
H3 LW=0.001	8	0.15625	1	14.29%
H4 LW=1	7	0.125	0	0.00%
H4 LW=10	7	0.125	0	0.00%
H4 LW=0.001	8	0.140625	1	14.29%
H5 LW=1	7	0.125	0	0.00%
H5 LW=10	7	0.140625	0	0.00%
H5 LW=0.001	8	0.125	1	14.29%
H6 LW=1	8	0.140625	1	14.29%
H6 LW=10	7	0.140625	0	0.00%
H6 LW=0.001	8	0.140625	1	14.29%
H7 LW=1	8	0.15625	1	14.29%
H7 LW=10	8	0.140625	1	14.29%
H7 LW=0.001	8	0.140625	1	14.29%
H8 LW=1	7	0.125	0	0.00%
H8 LW=10	7	0.15625	0	0.00%
H8 LW=0.001	8	0.125	1	14.29%
H9 LW=1	8	0.140625	1	14.29%
H9 LW=10	7	0.125	0	0.00%
H9 LW=0.001	8	0.125	1	14.29%
H10 LW=1	8	0.140625	1	14.29%
H10 LW=10	8	0.171875	1	14.29%
H10 LW=0.001	9	0.171875	2	28.57%
H11 LW=1	7	0.015625	0	0.00%
H11 LW=10	7	0.03125	0	0.00%
H11 LW=0.001	7	0.015625	0	0.00%
H12 LW=1	7	0.078125	0	0.00%
H12 LW=10	7	0.09375	0	0.00%

H12	LW=0.001	8	0.09375	1	14.29%
H13	LW=1	7	0.078125	0	0.00%
H13	LW=10	7	0.09375	0	0.00%
H13	LW=0.001	8	0.09375	1	14.29%
H14	LW=1	7	0.015625	0	0.00%
H14	LW=10	7	0.015625	0	0.00%
H14	LW=0.001	7	0.015625	0	0.00%
H15	LW=1	7	0.078125	0	0.00%
H15	LW=10	7	0.078125	0	0.00%
H15	LW=0.001	8	0.109375	1	14.29%
H16	LW=1	7	0.0625	0	0.00%
H16	LW=10	7	0.078125	0	0.00%
H16	LW=0.001	8	0.09375	1	14.29%
H17	LW=1	7	0.015625	0	0.00%
H17	LW=10	7	0.015625	0	0.00%
H17	LW=0.001	7	0.015625	0	0.00%
H18	LW=1	7	0.078125	0	0.00%
H18	LW=10	7	0.09375	0	0.00%
H18	LW=0.001	8	0.09375	1	14.29%
H19	LW=1	7	0.078125	0	0.00%
H19	LW=10	7	0.078125	0	0.00%
H19	LW=0.001	8	0.09375	1	14.29%
H20	LW=1	8	0.015625	1	14.29%
H20	LW=10	8	0.015625	1	14.29%
H20	LW=0.001	8	0.015625	1	14.29%
H21	LW=1	7	0.09375	0	0.00%
H21	LW=10	7	0.078125	0	0.00%
H21	LW=0.001	8	0.09375	1	14.29%
H22	LW=1	7	0.078125	0	0.00%
H22	LW=10	7	0.078125	0	0.00%
H22	LW=0.001	8	0.078125	1	14.29%
H23	LW=1	8	0.015625	1	14.29%
H23	LW=10	8	0.015625	1	14.29%
H23	LW=0.001	8	0.015625	1	14.29%
H24	LW=1	7	0.09375	0	0.00%
H24	LW=10	7	0.09375	0	0.00%
H24	LW=0.001	8	0.21875	1	14.29%
H25	LW=1	7	0.078125	0	0.00%
H25	LW=10	7	0.09375	0	0.00%
H25	LW=0.001	8	0.09375	1	14.29%
H26	LW=1	8	0.015625	1	14.29%
H26	LW=10	8	0.015625	1	14.29%
H26	LW=0.001	8	0.03125	1	14.29%
H27	LW=1	7	0.078125	0	0.00%
H27	LW=10	7	0.09375	0	0.00%
H27	LW=0.001	8	0.09375	1	14.29%
H28	LW=1	7	0.078125	0	0.00%

H28	LW=10	7	0.09375	0	0.00%
H28	LW=0.001	8	0.078125	1	14.29%

Test Instances		T2			
Job Number		3			
Operation Number		10			
Edge Number		227			
Node Number		35			
Total Length		15.5			
Average Length		1.55			
Optimum Time Slot		8			
	LWs	Makespan	Computation Time (s)	Difference	Error Rate
IP		8	68.6741747	0	0.00%
H1	LW=1	9	1.078125	1	12.50%
H1	LW=15.5	9	1.09375	1	12.50%
H1	LW=0.001	9	1.078125	1	12.50%
H2	LW=1	9	1.078125	1	12.50%
H2	LW=15.5	9	1.015625	1	12.50%
H2	LW=0.001	8	1.078125	0	0.00%
H3	LW=1	9	0.984375	1	12.50%
H3	LW=15.5	9	1.0625	1	12.50%
H3	LW=0.001	9	1.28125	1	12.50%
H4	LW=1	9	0.953125	1	12.50%
H4	LW=15.5	9	0.984375	1	12.50%
H4	LW=0.001	9	0.953125	1	12.50%
H5	LW=1	9	0.96875	1	12.50%
H5	LW=15.5	9	1	1	12.50%
H5	LW=0.001	10	1.1875	2	25.00%
H6	LW=1	8	1.03125	0	0.00%
H6	LW=15.5	9	1.078125	1	12.50%
H6	LW=0.001	9	1.046875	1	12.50%
H7	LW=1	8	1.0625	0	0.00%
H7	LW=15.5	8	1.0625	0	0.00%
H7	LW=0.001	8	1.125	0	0.00%
H8	LW=1	9	0.953125	1	12.50%
H8	LW=15.5	9	0.96875	1	12.50%
H8	LW=0.001	8	0.984375	0	0.00%
H9	LW=1	9	1.015625	1	12.50%
H9	LW=15.5	9	0.984375	1	12.50%

H9	LW=0.001	9	1.046875	1	12.50%
H10	LW=1	9	0.953125	1	12.50%
H10	LW=15.5	9	0.953125	1	12.50%
H10	LW=0.001	9	1.046875	1	12.50%
H11	LW=1	8	0.046875	0	0.00%
H11	LW=15.5	9	0.046875	1	12.50%
H11	LW=0.001	11	0.046875	3	37.50%
H12	LW=1	8	0.421875	0	0.00%
H12	LW=15.5	9	0.3125	1	12.50%
H12	LW=0.001	9	0.296875	1	12.50%
H13	LW=1	9	0.21875	1	12.50%
H13	LW=15.5	9	0.234375	1	12.50%
H13	LW=0.001	9	0.21875	1	12.50%
H14	LW=1	8	0.046875	0	0.00%
H14	LW=15.5	8	0.046875	0	0.00%
H14	LW=0.001	8	0.046875	0	0.00%
H15	LW=1	9	0.3125	1	12.50%
H15	LW=15.5	9	0.296875	1	12.50%
H15	LW=0.001	9	0.328125	1	12.50%
H16	LW=1	9	0.21875	1	12.50%
H16	LW=15.5	9	0.21875	1	12.50%
H16	LW=0.001	8	0.234375	0	0.00%
H17	LW=1	9	0.046875	1	12.50%
H17	LW=15.5	9	0.046875	1	12.50%
H17	LW=0.001	8	0.046875	0	0.00%
H18	LW=1	9	0.4375	1	12.50%
H18	LW=15.5	9	0.265625	1	12.50%
H18	LW=0.001	9	0.359375	1	12.50%
H19	LW=1	9	0.234375	1	12.50%
H19	LW=15.5	9	0.25	1	12.50%
H19	LW=0.001	8	0.265625	0	0.00%
H20	LW=1	9	0.046875	1	12.50%
H20	LW=15.5	9	0.046875	1	12.50%
H20	LW=0.001	9	0.046875	1	12.50%
H21	LW=1	9	0.296875	1	12.50%
H21	LW=15.5	9	0.265625	1	12.50%
H21	LW=0.001	9	0.25	1	12.50%
H22	LW=1	9	0.265625	1	12.50%
H22	LW=15.5	9	0.21875	1	12.50%
H22	LW=0.001	9	0.21875	1	12.50%
H23	LW=1	9	0.046875	1	12.50%
H23	LW=15.5	9	0.046875	1	12.50%

H23	LW=0.001	9	0.046875	1	12.50%
H24	LW=1	9	0.25	1	12.50%
H24	LW=15.5	9	0.28125	1	12.50%
H24	LW=0.001	8	0.296875	0	0.00%
H25	LW=1	9	0.234375	1	12.50%
H25	LW=15.5	9	0.265625	1	12.50%
H25	LW=0.001	8	0.265625	0	0.00%
H26	LW=1	9	0.0625	1	12.50%
H26	LW=15.5	9	0.046875	1	12.50%
H26	LW=0.001	9	0.046875	1	12.50%
H27	LW=1	9	0.265625	1	12.50%
H27	LW=15.5	9	0.234375	1	12.50%
H27	LW=0.001	8	0.28125	0	0.00%
H28	LW=1	9	0.21875	1	12.50%
H28	LW=15.5	9	0.28125	1	12.50%
H28	LW=0.001	8	0.265625	0	0.00%

Test Instances			T103		
Job Number			3		
Operation Number			11		
Edge Number			307		
Node Number			37		
Total Length			22		
Average Length			2		
Optimum Time Slot			16		
	LWs	Makespan	Computation Time (s)	Difference	Error Rate
IP		16	26.96629721	0	0.00%
H1	LW=1	16	1.28125	0	0.00%
H1	LW=22	16	1.296875	0	0.00%
H1	LW=0.001	19	2.25	3	18.75%
H2	LW=1	16	1.28125	0	0.00%
H2	LW=22	16	1.28125	0	0.00%
H2	LW=0.001	19	1.296875	3	18.75%
H3	LW=1	16	1.234375	0	0.00%
H3	LW=22	16	1.15625	0	0.00%
H3	LW=0.001	19	1.25	3	18.75%
H4	LW=1	16	1.296875	0	0.00%
H4	LW=22	16	1.109375	0	0.00%
H4	LW=0.001	17	2.125	1	6.25%

H5	LW=1	16	1.09375	0	0.00%
H5	LW=22	16	1.375	0	0.00%
H5	LW=0.001	17	1.265625	1	6.25%
H6	LW=1	18	1.015625	2	12.50%
H6	LW=22	18	1.0625	2	12.50%
H6	LW=0.001	19	1.109375	3	18.75%
H7	LW=1	18	1.203125	2	12.50%
H7	LW=22	18	1.203125	2	12.50%
H7	LW=0.001	18	1.15625	2	12.50%
H8	LW=1	16	1.390625	0	0.00%
H8	LW=22	16	1.109375	0	0.00%
H8	LW=0.001	17	1.203125	1	6.25%
H9	LW=1	18	1.3125	2	12.50%
H9	LW=22	18	1.1875	2	12.50%
H9	LW=0.001	19	1.140625	3	18.75%
H10	LW=1	18	1.0625	2	12.50%
H10	LW=22	18	1.078125	2	12.50%
H10	LW=0.001	19	1.140625	3	18.75%
H11	LW=1	16	0.078125	0	0.00%
H11	LW=22	16	0.078125	0	0.00%
H11	LW=0.001	17	0.078125	1	6.25%
H12	LW=1	16	0.546875	0	0.00%
H12	LW=22	16	0.515625	0	0.00%
H12	LW=0.001	19	0.625	3	18.75%
H13	LW=1	16	0.390625	0	0.00%
H13	LW=22	16	0.34375	0	0.00%
H13	LW=0.001	19	0.53125	3	18.75%
H14	LW=1	16	0.0625	0	0.00%
H14	LW=22	16	0.078125	0	0.00%
H14	LW=0.001	16	0.0625	0	0.00%
H15	LW=1	16	0.53125	0	0.00%
H15	LW=22	16	0.5	0	0.00%
H15	LW=0.001	19	0.46875	3	18.75%
H16	LW=1	16	0.359375	0	0.00%
H16	LW=22	16	0.40625	0	0.00%
H16	LW=0.001	19	0.40625	3	18.75%
H17	LW=1	16	0.0625	0	0.00%
H17	LW=22	16	0.09375	0	0.00%
H17	LW=0.001	16	0.0625	0	0.00%
H18	LW=1	16	0.46875	0	0.00%
H18	LW=22	16	0.546875	0	0.00%
H18	LW=0.001	19	0.5625	3	18.75%

H19	LW=1	16	0.375	0	0.00%
H19	LW=22	16	0.375	0	0.00%
H19	LW=0.001	19	0.40625	3	18.75%
H20	LW=1	18	0.09375	2	12.50%
H20	LW=22	18	0.15625	2	12.50%
H20	LW=0.001	18	0.078125	2	12.50%
H21	LW=1	16	0.4375	0	0.00%
H21	LW=22	16	0.40625	0	0.00%
H21	LW=0.001	19	0.609375	3	18.75%
H22	LW=1	16	0.359375	0	0.00%
H22	LW=22	16	0.375	0	0.00%
H22	LW=0.001	19	0.609375	3	18.75%
H23	LW=1	18	0.078125	2	12.50%
H23	LW=22	18	0.09375	2	12.50%
H23	LW=0.001	18	0.09375	2	12.50%
H24	LW=1	16	0.484375	0	0.00%
H24	LW=22	16	0.390625	0	0.00%
H24	LW=0.001	19	0.453125	3	18.75%
H25	LW=1	16	0.34375	0	0.00%
H25	LW=22	16	0.359375	0	0.00%
H25	LW=0.001	19	0.421875	3	18.75%
H26	LW=1	18	0.078125	2	12.50%
H26	LW=22	18	0.078125	2	12.50%
H26	LW=0.001	18	0.09375	2	12.50%
H27	LW=1	16	0.421875	0	0.00%
H27	LW=22	16	0.453125	0	0.00%
H27	LW=0.001	19	0.5	3	18.75%
H28	LW=1	16	0.375	0	0.00%
H28	LW=22	16	0.390625	0	0.00%
H28	LW=0.001	19	0.390625	3	18.75%

Test Instances	T4
Job Number	4
Operation Number	12
Edge Number	315
Node Number	41
Total Length	20.5
Average Length	1.708333333
Optimum Time Slot	11

	LWs	Makespan	Computation Time (s)	Difference	Error Rate
IP		11	118.9429157	0	0.00%
H1	LW=1	11	1.859375	0	0.00%
H1	LW=20.5	11	2.140625	0	0.00%
H1	LW=0.001	14	2.09375	3	27.27%
H2	LW=1	11	1.75	0	0.00%
H2	LW=20.5	11	1.75	0	0.00%
H2	LW=0.001	14	2.171875	3	27.27%
H3	LW=1	11	2.015625	0	0.00%
H3	LW=20.5	11	1.75	0	0.00%
H3	LW=0.001	14	2.046875	3	27.27%
H4	LW=1	11	1.90625	0	0.00%
H4	LW=20.5	11	1.890625	0	0.00%
H4	LW=0.001	14	2.03125	3	27.27%
H5	LW=1	11	1.96875	0	0.00%
H5	LW=20.5	11	1.90625	0	0.00%
H5	LW=0.001	14	2	3	27.27%
H6	LW=1	13	1.9375	2	18.18%
H6	LW=20.5	13	1.84375	2	18.18%
H6	LW=0.001	14	1.953125	3	27.27%
H7	LW=1	13	2.140625	2	18.18%
H7	LW=20.5	13	2.1875	2	18.18%
H7	LW=0.001	14	2.109375	3	27.27%
H8	LW=1	11	1.78125	0	0.00%
H8	LW=20.5	11	1.84375	0	0.00%
H8	LW=0.001	14	2.078125	3	27.27%
H9	LW=1	12	1.78125	1	9.09%
H9	LW=20.5	13	1.953125	2	18.18%
H9	LW=0.001	14	2.078125	3	27.27%
H10	LW=1	12	1.921875	1	9.09%
H10	LW=20.5	13	1.859375	2	18.18%
H10	LW=0.001	14	1.984375	3	27.27%
H11	LW=1	12	0.09375	1	9.09%
H11	LW=20.5	15	0.203125	4	36.36%
H11	LW=0.001	14	0.078125	3	27.27%
H12	LW=1	12	0.375	1	9.09%
H12	LW=20.5	13	0.4375	2	18.18%
H12	LW=0.001	14	0.578125	3	27.27%
H13	LW=1	11	0.359375	0	0.00%
H13	LW=20.5	11	0.390625	0	0.00%
H13	LW=0.001	14	0.421875	3	27.27%
H14	LW=1	12	0.078125	1	9.09%
H14	LW=20.5	13	0.09375	2	18.18%

H14	LW=0.001	14	0.09375	3	27.27%
H15	LW=1	12	0.453125	1	9.09%
H15	LW=20.5	13	0.453125	2	18.18%
H15	LW=0.001	14	0.640625	3	27.27%
H16	LW=1	11	0.34375	0	0.00%
H16	LW=20.5	11	0.390625	0	0.00%
H16	LW=0.001	14	0.4375	3	27.27%
H17	LW=1	12	0.078125	1	9.09%
H17	LW=20.5	13	0.09375	2	18.18%
H17	LW=0.001	14	0.078125	3	27.27%
H18	LW=1	13	0.453125	2	18.18%
H18	LW=20.5	13	0.546875	2	18.18%
H18	LW=0.001	14	0.671875	3	27.27%
H19	LW=1	11	0.375	0	0.00%
H19	LW=20.5	11	0.34375	0	0.00%
H19	LW=0.001	14	0.4375	3	27.27%
H20	LW=1	14	0.09375	3	27.27%
H20	LW=20.5	14	0.09375	3	27.27%
H20	LW=0.001	14	0.09375	3	27.27%
H21	LW=1	11	0.40625	0	0.00%
H21	LW=20.5	11	0.5	0	0.00%
H21	LW=0.001	13	0.40625	2	18.18%
H22	LW=1	11	0.3125	0	0.00%
H22	LW=20.5	11	0.3125	0	0.00%
H22	LW=0.001	14	0.390625	3	27.27%
H23	LW=1	14	0.09375	3	27.27%
H23	LW=20.5	14	0.109375	3	27.27%
H23	LW=0.001	13	0.078125	2	18.18%
H24	LW=1	11	0.4375	0	0.00%
H24	LW=20.5	11	0.421875	0	0.00%
H24	LW=0.001	13	0.421875	2	18.18%
H25	LW=1	11	0.3125	0	0.00%
H25	LW=20.5	11	0.328125	0	0.00%
H25	LW=0.001	14	0.375	3	27.27%
H26	LW=1	14	0.09375	3	27.27%
H26	LW=20.5	14	0.109375	3	27.27%
H26	LW=0.001	13	0.0625	2	18.18%
H27	LW=1	11	0.421875	0	0.00%
H27	LW=20.5	11	0.46875	0	0.00%
H27	LW=0.001	13	0.421875	2	18.18%
H28	LW=1	11	0.359375	0	0.00%
H28	LW=20.5	11	0.328125	0	0.00%
H28	LW=0.001	14	0.421875	3	27.27%

Test Instances		T5			
Job Number		4			
Operation Number		12			
Edge Number		390			
Node Number		41			
Total Length		20.5			
Average Length		1.708333333			
Optimum Time Slot		10			
	LWs	Makespan	Computation Time (s)	Difference	Error Rate
IP		10	150.7721189	0	0.00%
H1	LW=1	10	2.296875	0	0.00%
H1	LW=20.5	10	2.78125	0	0.00%
H1	LW=0.001	10	2.421875	0	0.00%
H2	LW=1	10	2.21875	0	0.00%
H2	LW=20.5	12	2.671875	2	200.00%
H2	LW=0.001	12	2.15625	2	200.00%
H3	LW=1	10	2.125	0	0.00%
H3	LW=20.5	12	2.171875	2	200.00%
H3	LW=0.001	10	2.1875	0	0.00%
H4	LW=1	10	2.109375	0	0.00%
H4	LW=20.5	10	2.078125	0	0.00%
H4	LW=0.001	10	2.21875	0	0.00%
H5	LW=1	10	2.25	0	0.00%
H5	LW=20.5	12	2.21875	2	200.00%
H5	LW=0.001	12	1.890625	2	200.00%
H6	LW=1	12	2.171875	2	200.00%
H6	LW=20.5	11	1.9375	1	100.00%
H6	LW=0.001	13	2.28125	3	300.00%
H7	LW=1	12	2.53125	2	200.00%
H7	LW=20.5	12	2.640625	2	200.00%
H7	LW=0.001	14	2.453125	4	400.00%
H8	LW=1	10	2.0625	0	0.00%
H8	LW=20.5	12	2.125	2	200.00%
H8	LW=0.001	10	2.265625	0	0.00%
H9	LW=1	11	2.0625	1	100.00%
H9	LW=20.5	11	1.859375	1	100.00%
H9	LW=0.001	13	2.125	3	300.00%
H10	LW=1	11	2	1	100.00%
H10	LW=20.5	11	1.921875	1	100.00%
H10	LW=0.001	13	2.078125	3	300.00%

H11	LW=1	13	0.109375	3	300.00%
H11	LW=20.5	11	0.109375	1	100.00%
H11	LW=0.001	11	0.09375	1	100.00%
H12	LW=1	14	0.90625	4	400.00%
H12	LW=20.5	11	0.5	1	100.00%
H12	LW=0.001	14	0.765625	4	400.00%
H13	LW=1	10	0.4375	0	0.00%
H13	LW=20.5	10	0.4375	0	0.00%
H13	LW=0.001	10	0.484375	0	0.00%
H14	LW=1	13	0.09375	3	300.00%
H14	LW=20.5	13	0.15625	3	300.00%
H14	LW=0.001	11	0.09375	1	100.00%
H15	LW=1	14	0.921875	4	400.00%
H15	LW=20.5	11	0.703125	1	100.00%
H15	LW=0.001	13	0.703125	3	300.00%
H16	LW=1	10	0.453125	0	0.00%
H16	LW=20.5	12	0.484375	2	200.00%
H16	LW=0.001	12	0.4375	2	200.00%
H17	LW=1	13	0.109375	3	300.00%
H17	LW=20.5	11	0.09375	1	100.00%
H17	LW=0.001	13	0.125	3	300.00%
H18	LW=1	14	0.8125	4	400.00%
H18	LW=20.5	11	0.625	1	100.00%
H18	LW=0.001	12	0.578125	2	200.00%
H19	LW=1	10	0.421875	0	0.00%
H19	LW=20.5	12	0.46875	2	200.00%
H19	LW=0.001	10	0.4375	0	0.00%
H20	LW=1	13	0.109375	3	300.00%
H20	LW=20.5	13	0.09375	3	300.00%
H20	LW=0.001	13	0.09375	3	300.00%
H21	LW=1	11	0.5625	1	100.00%
H21	LW=20.5	11	0.53125	1	100.00%
H21	LW=0.001	12	0.484375	2	200.00%
H22	LW=1	10	0.375	0	0.00%
H22	LW=20.5	10	0.390625	0	0.00%
H22	LW=0.001	10	0.40625	0	0.00%
H23	LW=1	13	0.09375	3	300.00%
H23	LW=20.5	13	0.109375	3	300.00%
H23	LW=0.001	13	0.109375	3	300.00%
H24	LW=1	12	0.546875	2	200.00%
H24	LW=20.5	11	0.765625	1	100.00%
H24	LW=0.001	12	0.5	2	200.00%
H25	LW=1	10	0.375	0	0.00%

H25	LW=20.5	12	0.40625	2	200.00%
H25	LW=0.001	12	0.375	2	200.00%
H26	LW=1	13	0.09375	3	300.00%
H26	LW=20.5	13	0.09375	3	300.00%
H26	LW=0.001	13	0.09375	3	300.00%
H27	LW=1	13	0.5625	3	300.00%
H27	LW=20.5	11	0.578125	1	100.00%
H27	LW=0.001	12	0.59375	2	200.00%
H28	LW=1	10	0.359375	0	0.00%
H28	LW=20.5	12	0.421875	2	200.00%
H28	LW=0.001	10	0.390625	0	0.00%

Test Instances		T6			
Job Number		5			
Operation Number		13			
Edge Number		316			
Node Number		40			
Total Length		17.5			
Average Length		1.346153846			
Optimum Time Slot		7			
	LWs	Makespan	Computation Time (s)	Difference	Error Rate
IP		7	643.7664479	0	0.00%
H1	LW=1	8	1.609375	1	14.29%
H1	LW=17.5	8	1.9375	1	14.29%
H1	LW=0.001	10	1.984375	3	42.86%
H2	LW=1	8	1.484375	1	14.29%
H2	LW=17.5	8	1.84375	1	14.29%
H2	LW=0.001	10	1.5625	3	42.86%
H3	LW=1	8	1.59375	1	14.29%
H3	LW=17.5	8	1.84375	1	14.29%
H3	LW=0.001	10	1.53125	3	42.86%
H4	LW=1	8	1.390625	1	14.29%
H4	LW=17.5	8	1.921875	1	14.29%
H4	LW=0.001	10	1.65625	3	42.86%
H5	LW=1	8	1.640625	1	14.29%
H5	LW=17.5	8	1.65625	1	14.29%
H5	LW=0.001	10	1.53125	3	42.86%
H6	LW=1	9	1.671875	2	28.57%
H6	LW=17.5	9	1.46875	2	28.57%
H6	LW=0.001	9	1.734375	2	28.57%

H7	LW=1	9	1.828125	2	28.57%
H7	LW=17.5	9	1.78125	2	28.57%
H7	LW=0.001	9	1.765625	2	28.57%
H8	LW=1	8	1.375	1	14.29%
H8	LW=17.5	8	1.65625	1	14.29%
H8	LW=0.001	10	1.390625	3	42.86%
H9	LW=1	9	1.515625	2	28.57%
H9	LW=17.5	9	1.5	2	28.57%
H9	LW=0.001	9	1.84375	2	28.57%
H10	LW=1	9	1.46875	2	28.57%
H10	LW=17.5	9	1.4375	2	28.57%
H10	LW=0.001	9	1.859375	2	28.57%
H11	LW=1	9	0.0625	2	28.57%
H11	LW=17.5	8	0.0625	1	14.29%
H11	LW=0.001	10	0.046875	3	42.86%
H12	LW=1	8	0.421875	1	14.29%
H12	LW=17.5	8	0.453125	1	14.29%
H12	LW=0.001	10	0.53125	3	42.86%
H13	LW=1	10	0.421875	3	42.86%
H13	LW=17.5	8	0.390625	1	14.29%
H13	LW=0.001	10	0.375	3	42.86%
H14	LW=1	9	0.0625	2	28.57%
H14	LW=17.5	8	0.09375	1	14.29%
H14	LW=0.001	8	0.0625	1	14.29%
H15	LW=1	8	0.5	1	14.29%
H15	LW=17.5	8	0.53125	1	14.29%
H15	LW=0.001	10	0.46875	3	42.86%
H16	LW=1	8	0.359375	1	14.29%
H16	LW=17.5	8	0.359375	1	14.29%
H16	LW=0.001	10	0.40625	3	42.86%
H17	LW=1	9	0.0625	2	28.57%
H17	LW=17.5	8	0.09375	1	14.29%
H17	LW=0.001	8	0.0625	1	14.29%
H18	LW=1	8	0.484375	1	14.29%
H18	LW=17.5	9	0.40625	2	28.57%
H18	LW=0.001	9	0.578125	2	28.57%
H19	LW=1	8	0.375	1	14.29%
H19	LW=17.5	8	0.375	1	14.29%
H19	LW=0.001	10	0.40625	3	42.86%
H20	LW=1	8	0.0625	1	14.29%
H20	LW=17.5	9	0.0625	2	28.57%
H20	LW=0.001	8	0.078125	1	14.29%
H21	LW=1	9	0.546875	2	28.57%

H21	LW=17.5	8	0.4375	1	14.29%
H21	LW=0.001	9	0.59375	2	28.57%
H22	LW=1	10	0.375	3	42.86%
H22	LW=17.5	8	0.328125	1	14.29%
H22	LW=0.001	10	0.375	3	42.86%
H23	LW=1	9	0.0625	2	28.57%
H23	LW=17.5	9	0.0625	2	28.57%
H23	LW=0.001	8	0.0625	1	14.29%
H24	LW=1	8	0.46875	1	14.29%
H24	LW=17.5	8	0.578125	1	14.29%
H24	LW=0.001	9	0.390625	2	28.57%
H25	LW=1	8	0.328125	1	14.29%
H25	LW=17.5	8	0.359375	1	14.29%
H25	LW=0.001	8	0.359375	1	14.29%
H26	LW=1	9	0.0625	2	28.57%
H26	LW=17.5	9	0.0625	2	28.57%
H26	LW=0.001	9	0.0625	2	28.57%
H27	LW=1	8	0.421875	1	14.29%
H27	LW=17.5	9	0.453125	2	28.57%
H27	LW=0.001	8	0.5	1	14.29%
H28	LW=1	8	0.578125	1	14.29%
H28	LW=17.5	8	0.359375	1	14.29%
H28	LW=0.001	10	0.390625	3	42.86%

Test Instances			T7		
Job Number			5		
Operation Number			14		
Edge Number			396		
Node Number			41		
Total Length			20.5		
Average Length			1.464285714		
Optimum Time Slot			10		
	LWs	Makespan	Computation Time (s)	Difference	Error Rate
IP		10	5794.495398	0	0.00%
H1	LW=1	10	1.90625	0	0.00%
H1	LW=20.5	10	1.96875	0	0.00%
H1	LW=0.001	10	1.921875	0	0.00%
H2	LW=1	10	1.890625	0	0.00%

H2	LW=20.5	10	1.8125	0	0.00%
H2	LW=0.001	12	1.734375	2	20.00%
H3	LW=1	10	1.8125	0	0.00%
H3	LW=20.5	10	1.921875	0	0.00%
H3	LW=0.001	10	1.9375	0	0.00%
H4	LW=1	10	1.828125	0	0.00%
H4	LW=20.5	10	1.859375	0	0.00%
H4	LW=0.001	10	1.859375	0	0.00%
H5	LW=1	10	1.8125	0	0.00%
H5	LW=20.5	10	1.84375	0	0.00%
H5	LW=0.001	12	1.625	2	20.00%
H6	LW=1	12	1.859375	2	20.00%
H6	LW=20.5	11	1.671875	1	10.00%
H6	LW=0.001	14	2.15625	4	40.00%
H7	LW=1	12	1.96875	2	20.00%
H7	LW=20.5	12	2.03125	2	20.00%
H7	LW=0.001	14	2.203125	4	40.00%
H8	LW=1	10	1.78125	0	0.00%
H8	LW=20.5	10	1.921875	0	0.00%
H8	LW=0.001	10	1.75	0	0.00%
H9	LW=1	11	1.59375	1	10.00%
H9	LW=20.5	11	1.671875	1	10.00%
H9	LW=0.001	13	2	3	30.00%
H10	LW=1	11	2.234375	1	10.00%
H10	LW=20.5	11	1.90625	1	10.00%
H10	LW=0.001	13	2.140625	3	30.00%
H11	LW=1	11	0.09375	1	10.00%
H11	LW=20.5	11	0.109375	1	10.00%
H11	LW=0.001	11	0.15625	1	10.00%
H12	LW=1	10	0.65625	0	0.00%
H12	LW=20.5	10	0.703125	0	0.00%
H12	LW=0.001	10	0.53125	0	0.00%
H13	LW=1	10	0.4375	0	0.00%
H13	LW=20.5	10	0.421875	0	0.00%
H13	LW=0.001	10	0.390625	0	0.00%
H14	LW=1	11	0.09375	1	10.00%
H14	LW=20.5	12	0.09375	2	20.00%
H14	LW=0.001	11	0.109375	1	10.00%
H15	LW=1	10	0.65625	0	0.00%
H15	LW=20.5	10	0.65625	0	0.00%
H15	LW=0.001	12	0.640625	2	20.00%
H16	LW=1	10	0.421875	0	0.00%
H16	LW=20.5	10	0.546875	0	0.00%

H16	LW=0.001	12	0.703125	2	20.00%
H17	LW=1	11	0.09375	1	10.00%
H17	LW=20.5	11	0.109375	1	10.00%
H17	LW=0.001	11	0.109375	1	10.00%
H18	LW=1	10	0.578125	0	0.00%
H18	LW=20.5	10	0.609375	0	0.00%
H18	LW=0.001	14	0.984375	4	40.00%
H19	LW=1	10	0.40625	0	0.00%
H19	LW=20.5	10	0.484375	0	0.00%
H19	LW=0.001	10	0.546875	0	0.00%
H20	LW=1	12	0.09375	2	20.00%
H20	LW=20.5	12	0.09375	2	20.00%
H20	LW=0.001	12	0.09375	2	20.00%
H21	LW=1	12	0.546875	2	20.00%
H21	LW=20.5	12	0.546875	2	20.00%
H21	LW=0.001	12	0.5	2	20.00%
H22	LW=1	10	0.40625	0	0.00%
H22	LW=20.5	10	0.390625	0	0.00%
H22	LW=0.001	10	0.421875	0	0.00%
H23	LW=1	12	0.109375	2	20.00%
H23	LW=20.5	12	0.09375	2	20.00%
H23	LW=0.001	12	0.109375	2	20.00%
H24	LW=1	12	0.609375	2	20.00%
H24	LW=20.5	12	0.5625	2	20.00%
H24	LW=0.001	12	0.5	2	20.00%
H25	LW=1	10	0.40625	0	0.00%
H25	LW=20.5	10	0.4375	0	0.00%
H25	LW=0.001	12	0.4375	2	20.00%
H26	LW=1	12	0.109375	2	20.00%
H26	LW=20.5	12	0.09375	2	20.00%
H26	LW=0.001	12	0.09375	2	20.00%
H27	LW=1	12	0.578125	2	20.00%
H27	LW=20.5	12	0.5	2	20.00%
H27	LW=0.001	12	0.59375	2	20.00%
H28	LW=1	10	0.484375	0	0.00%
H28	LW=20.5	10	0.40625	0	0.00%
H28	LW=0.001	10	0.421875	0	0.00%

Test Instances	T8
Job Number	5

Operation Number		14			
Edge Number		504			
Node Number		45			
Total Length		20.5			
Average Length		1.464285714			
Optimum Time Slot		10			
	LWs	Makespan	Computation Time (s)	Difference	Error Rate
IP		10	12196.93827	0	0.00%
H1	LW=1	10	3.8125	0	0.00%
H1	LW=20.5	10	3.546875	0	0.00%
H1	LW=0.001	10	3.5625	0	0.00%
H2	LW=1	12	3.109375	2	20.00%
H2	LW=20.5	10	3.21875	0	0.00%
H2	LW=0.001	12	3.890625	2	20.00%
H3	LW=1	10	3.546875	0	0.00%
H3	LW=20.5	10	3.671875	0	0.00%
H3	LW=0.001	10	3.703125	0	0.00%
H4	LW=1	10	3.609375	0	0.00%
H4	LW=20.5	10	3.625	0	0.00%
H4	LW=0.001	10	3.390625	0	0.00%
H5	LW=1	12	3.375	2	20.00%
H5	LW=20.5	10	3.3125	0	0.00%
H5	LW=0.001	13	3.984375	3	30.00%
H6	LW=1	12	3.578125	2	20.00%
H6	LW=20.5	11	3.015625	1	10.00%
H6	LW=0.001	14	3.90625	4	40.00%
H7	LW=1	12	3.84375	2	20.00%
H7	LW=20.5	12	3.75	2	20.00%
H7	LW=0.001	12	3.78125	2	20.00%
H8	LW=1	10	3.578125	0	0.00%
H8	LW=20.5	10	3.296875	0	0.00%
H8	LW=0.001	10	3.484375	0	0.00%
H9	LW=1	11	3.21875	1	10.00%
H9	LW=20.5	11	2.984375	1	10.00%
H9	LW=0.001	13	4.046875	3	30.00%
H10	LW=1	11	3.1875	1	10.00%
H10	LW=20.5	11	2.96875	1	10.00%
H10	LW=0.001	13	4.0625	3	30.00%
H11	LW=1	11	0.140625	1	10.00%
H11	LW=20.5	11	0.125	1	10.00%
H11	LW=0.001	11	0.171875	1	10.00%
H12	LW=1	11	0.96875	1	10.00%
H12	LW=20.5	11	0.984375	1	10.00%
H12	LW=0.001	12	1.09375	2	20.00%

H13	LW=1	10	0.671875	0	0.00%
H13	LW=20.5	10	0.703125	0	0.00%
H13	LW=0.001	10	0.75	0	0.00%
H14	LW=1	11	0.15625	1	10.00%
H14	LW=20.5	11	0.140625	1	10.00%
H14	LW=0.001	14	0.15625	4	40.00%
H15	LW=1	13	1.34375	3	30.00%
H15	LW=20.5	11	0.875	1	10.00%
H15	LW=0.001	12	0.90625	2	20.00%
H16	LW=1	12	0.703125	2	20.00%
H16	LW=20.5	10	0.6875	0	0.00%
H16	LW=0.001	12	0.828125	2	20.00%
H17	LW=1	11	0.140625	1	10.00%
H17	LW=20.5	11	0.140625	1	10.00%
H17	LW=0.001	11	0.15625	1	10.00%
H18	LW=1	11	1.296875	1	10.00%
H18	LW=20.5	11	0.828125	1	10.00%
H18	LW=0.001	11	1	1	10.00%
H19	LW=1	10	0.703125	0	0.00%
H19	LW=20.5	10	0.671875	0	0.00%
H19	LW=0.001	10	0.703125	0	0.00%
H20	LW=1	12	0.140625	2	20.00%
H20	LW=20.5	12	0.15625	2	20.00%
H20	LW=0.001	12	0.125	2	20.00%
H21	LW=1	11	0.78125	1	10.00%
H21	LW=20.5	11	0.84375	1	10.00%
H21	LW=0.001	12	0.953125	2	20.00%
H22	LW=1	10	0.625	0	0.00%
H22	LW=20.5	10	0.75	0	0.00%
H22	LW=0.001	10	0.625	0	0.00%
H23	LW=1	12	0.140625	2	20.00%
H23	LW=20.5	12	0.140625	2	20.00%
H23	LW=0.001	12	0.125	2	20.00%
H24	LW=1	12	0.90625	2	20.00%
H24	LW=20.5	11	0.78125	1	10.00%
H24	LW=0.001	12	0.984375	2	20.00%
H25	LW=1	12	0.625	2	20.00%
H25	LW=20.5	10	0.65625	0	0.00%
H25	LW=0.001	12	0.703125	2	20.00%
H26	LW=1	12	0.15625	2	20.00%
H26	LW=20.5	12	0.140625	2	20.00%
H26	LW=0.001	12	0.125	2	20.00%
H27	LW=1	10	0.828125	0	0.00%
H27	LW=20.5	11	0.734375	1	10.00%

H27	LW=0.001	10	0.90625	0	0.00%
H28	LW=1	10	0.703125	0	0.00%
H28	LW=20.5	10	0.625	0	0.00%
H28	LW=0.001	10	0.71875	0	0.00%

Test Instances		T9			
Job Number		4			
Operation Number		14			
Edge Number		375			
Node Number		41			
Total Length		25			
Average Length		1.785714286			
Optimum Time Slot		18			
	LWs	Makespan	Computation Time (s)	Difference	Error Rate
IP		18	213.9238512	0	0.00%
H1	LW=1	18	1.71875	0	0.00%
H1	LW=25	18	1.71875	0	0.00%
H1	LW=0.001	20	3.046875	2	11.11%
H2	LW=1	18	1.65625	0	0.00%
H2	LW=25	18	1.609375	0	0.00%
H2	LW=0.001	20	2.328125	2	11.11%
H3	LW=1	19	1.625	1	5.56%
H3	LW=25	18	1.671875	0	0.00%
H3	LW=0.001	20	1.640625	2	11.11%
H4	LW=1	18	1.546875	0	0.00%
H4	LW=25	18	1.59375	0	0.00%
H4	LW=0.001	20	2.4375	2	11.11%
H5	LW=1	18	1.671875	0	0.00%
H5	LW=25	18	1.65625	0	0.00%
H5	LW=0.001	20	1.9375	2	11.11%
H6	LW=1	19	1.40625	1	5.56%
H6	LW=25	19	1.484375	1	5.56%
H6	LW=0.001	20	1.4375	2	11.11%
H7	LW=1	19	1.640625	1	5.56%
H7	LW=25	19	1.6875	1	5.56%
H7	LW=0.001	19	1.609375	1	5.56%
H8	LW=1	18	1.6875	0	0.00%
H8	LW=25	18	1.578125	0	0.00%
H8	LW=0.001	18	1.578125	0	0.00%
H9	LW=1	19	1.421875	1	5.56%

H9	LW=25	19	1.421875	1	5.56%
H9	LW=0.001	20	1.390625	2	11.11%
H10	LW=1	19	1.359375	1	5.56%
H10	LW=25	19	1.375	1	5.56%
H10	LW=0.001	20	1.40625	2	11.11%
H11	LW=1	18	0.109375	0	0.00%
H11	LW=25	18	0.109375	0	0.00%
H11	LW=0.001	18	0.15625	0	0.00%
H12	LW=1	18	0.6875	0	0.00%
H12	LW=25	18	0.859375	0	0.00%
H12	LW=0.001	18	0.859375	0	0.00%
H13	LW=1	18	0.59375	0	0.00%
H13	LW=25	18	0.578125	0	0.00%
H13	LW=0.001	20	0.734375	2	11.11%
H14	LW=1	18	0.109375	0	0.00%
H14	LW=25	18	0.125	0	0.00%
H14	LW=0.001	18	0.140625	0	0.00%
H15	LW=1	18	0.734375	0	0.00%
H15	LW=25	18	0.75	0	0.00%
H15	LW=0.001	18	0.828125	0	0.00%
H16	LW=1	18	0.65625	0	0.00%
H16	LW=25	18	0.546875	0	0.00%
H16	LW=0.001	20	0.65625	2	11.11%
H17	LW=1	18	0.125	0	0.00%
H17	LW=25	18	0.109375	0	0.00%
H17	LW=0.001	18	0.109375	0	0.00%
H18	LW=1	18	0.78125	0	0.00%
H18	LW=25	18	0.75	0	0.00%
H18	LW=0.001	19	0.6875	1	5.56%
H19	LW=1	19	0.59375	1	5.56%
H19	LW=25	18	0.578125	0	0.00%
H19	LW=0.001	20	0.65625	2	11.11%
H20	LW=1	20	0.140625	2	11.11%
H20	LW=25	20	0.140625	2	11.11%
H20	LW=0.001	20	0.140625	2	11.11%
H21	LW=1	18	0.625	0	0.00%
H21	LW=25	18	0.703125	0	0.00%
H21	LW=0.001	20	0.9375	2	11.11%
H22	LW=1	18	0.5625	0	0.00%
H22	LW=25	18	0.546875	0	0.00%
H22	LW=0.001	20	0.6875	2	11.11%
H23	LW=1	20	0.140625	2	11.11%
H23	LW=25	20	0.140625	2	11.11%

H23	LW=0.001	20	0.140625	2	11.11%
H24	LW=1	18	0.625	0	0.00%
H24	LW=25	18	0.671875	0	0.00%
H24	LW=0.001	20	0.78125	2	11.11%
H25	LW=1	18	0.59375	0	0.00%
H25	LW=25	18	0.546875	0	0.00%
H25	LW=0.001	20	0.625	2	11.11%
H26	LW=1	20	0.140625	2	11.11%
H26	LW=25	20	0.140625	2	11.11%
H26	LW=0.001	20	0.140625	2	11.11%
H27	LW=1	19	0.734375	1	5.56%
H27	LW=25	18	0.65625	0	0.00%
H27	LW=0.001	20	0.765625	2	11.11%
H28	LW=1	19	0.578125	1	5.56%
H28	LW=25	18	0.53125	0	0.00%
H28	LW=0.001	20	0.59375	2	11.11%

Test Instances			T10		
Job Number			4		
Operation Number			14		
Edge Number			1074		
Node Number			63		
Total Length			25		
Average Length			1.785714286		
Optimum Time Slot			18		
	LWs	Makespan	Computation Time (s)	Difference	Error Rate
IP		18	2959.314596	0	0.00%
H1	LW=1	18	10.21875	0	0.00%
H1	LW=25	18	10.484375	0	0.00%
H1	LW=0.001	20	8.453125	2	11.11%
H2	LW=1	18	10.171875	0	0.00%
H2	LW=25	18	10.265625	0	0.00%
H2	LW=0.001	20	8.078125	2	11.11%
H3	LW=1	18	10.40625	0	0.00%
H3	LW=25	18	9.953125	0	0.00%
H3	LW=0.001	20	11.296875	2	11.11%
H4	LW=1	18	7.890625	0	0.00%
H4	LW=25	18	7.71875	0	0.00%
H4	LW=0.001	20	6.609375	2	11.11%
H5	LW=1	18	7.8125	0	0.00%

H5	LW=25	18	7.546875	0	0.00%
H5	LW=0.001	20	6.703125	2	11.11%
H6	LW=1	18	8.484375	0	0.00%
H6	LW=25	18	8.453125	0	0.00%
H6	LW=0.001	20	8.78125	2	11.11%
H7	LW=1	19	9.828125	1	5.56%
H7	LW=25	19	9.953125	1	5.56%
H7	LW=0.001	19	9.78125	1	5.56%
H8	LW=1	18	7.859375	0	0.00%
H8	LW=25	18	8.203125	0	0.00%
H8	LW=0.001	18	8.46875	0	0.00%
H9	LW=1	19	8.203125	1	5.56%
H9	LW=25	18	8.25	0	0.00%
H9	LW=0.001	20	8.671875	2	11.11%
H10	LW=1	19	8.28125	1	5.56%
H10	LW=25	19	8.6875	1	5.56%
H10	LW=0.001	20	8.78125	2	11.11%
H11	LW=1	18	0.625	0	0.00%
H11	LW=25	18	0.640625	0	0.00%
H11	LW=0.001	18	0.515625	0	0.00%
H12	LW=1	18	3.609375	0	0.00%
H12	LW=25	18	3.890625	0	0.00%
H12	LW=0.001	20	4.171875	2	11.11%
H13	LW=1	18	3.078125	0	0.00%
H13	LW=25	18	3.09375	0	0.00%
H13	LW=0.001	20	3.609375	2	11.11%
H14	LW=1	18	0.640625	0	0.00%
H14	LW=25	18	0.609375	0	0.00%
H14	LW=0.001	18	0.65625	0	0.00%
H15	LW=1	18	3.671875	0	0.00%
H15	LW=25	18	3.5	0	0.00%
H15	LW=0.001	20	4.046875	2	11.11%
H16	LW=1	18	3.375	0	0.00%
H16	LW=25	18	3.046875	0	0.00%
H16	LW=0.001	20	3.609375	2	11.11%
H17	LW=1	18	0.640625	0	0.00%
H17	LW=25	18	0.640625	0	0.00%
H17	LW=0.001	18	0.734375	0	0.00%
H18	LW=1	18	3.796875	0	0.00%
H18	LW=25	18	3.59375	0	0.00%
H18	LW=0.001	20	4.03125	2	11.11%
H19	LW=1	18	3.5	0	0.00%
H19	LW=25	18	3.203125	0	0.00%

H19	LW=0.001	20	3.59375	2	11.11%
H20	LW=1	19	0.78125	1	5.56%
H20	LW=25	19	0.703125	1	5.56%
H20	LW=0.001	19	0.6875	1	5.56%
H21	LW=1	18	3.71875	0	0.00%
H21	LW=25	18	3.84375	0	0.00%
H21	LW=0.001	20	4.65625	2	11.11%
H22	LW=1	18	3.0625	0	0.00%
H22	LW=25	18	3.1875	0	0.00%
H22	LW=0.001	20	3.390625	2	11.11%
H23	LW=1	19	0.796875	1	5.56%
H23	LW=25	19	0.703125	1	5.56%
H23	LW=0.001	19	0.671875	1	5.56%
H24	LW=1	18	3.65625	0	0.00%
H24	LW=25	18	3.6875	0	0.00%
H24	LW=0.001	20	4.015625	2	11.11%
H25	LW=1	18	3.234375	0	0.00%
H25	LW=25	18	3.046875	0	0.00%
H25	LW=0.001	20	3.53125	2	11.11%
H26	LW=1	19	0.6875	1	5.56%
H26	LW=25	19	0.671875	1	5.56%
H26	LW=0.001	19	0.71875	1	5.56%
H27	LW=1	18	3.703125	0	0.00%
H27	LW=25	18	3.53125	0	0.00%
H27	LW=0.001	20	3.90625	2	11.11%
H28	LW=1	18	3.359375	0	0.00%
H28	LW=25	18	3.234375	0	0.00%
H28	LW=0.001	20	3.578125	2	11.11%

Test Instances		T11			
Job Number		4			
Operation Number		14			
Edge Number		4718			
Node Number		161			
Total Length		32			
Average Length		2.285714286			
Optimum Time Slot		10			
	LWs	Makespan	Computation Time (s)	Difference	Error Rate
IP		10		0	0.00%
H1	LW=1	13	11553.51563	3	30.00%

H1	LW=32	11	13094.35938	1	10.00%
H1	LW=0.001	15	10657.78125	5	50.00%
H2	LW=1	11	14337.90625	1	10.00%
H2	LW=32	11	14001.375	1	10.00%
H2	LW=0.001	16	14838.57813	6	60.00%
H3	LW=1	11	14707.34375	1	10.00%
H3	LW=32	11	12879.375	1	10.00%
H3	LW=0.001	15	12523.5	5	50.00%
H4	LW=1	13	10329.75	3	30.00%
H4	LW=32	11	14309.79688	1	10.00%
H4	LW=0.001	16	15394.65625	6	60.00%
H5	LW=1	11	9859.75	1	10.00%
H5	LW=32	11	13247.4375	1	10.00%
H5	LW=0.001	16	9605.890625	6	60.00%
H6	LW=1	11	12772.60938	1	10.00%
H6	LW=32	11	11499.90625	1	10.00%
H6	LW=0.001	16	11853.82813	6	60.00%
H7	LW=1	14	11999.90625	4	40.00%
H7	LW=32	14	12030.14063	4	40.00%
H7	LW=0.001	16	11798.59375	6	60.00%
H8	LW=1	11	10411.01563	1	10.00%
H8	LW=32	11	10112.26563	1	10.00%
H8	LW=0.001	15	10013.32813	5	50.00%
H9	LW=1	11	10395.29688	1	10.00%
H9	LW=32	11	10427.29688	1	10.00%
H9	LW=0.001	15	10665.67188	5	50.00%
H10	LW=1	11	11059.375	1	10.00%
H10	LW=32	11	10856.64063	1	10.00%
H10	LW=0.001	16	10585.51563	6	60.00%
H11	LW=1	14	8.828125	4	40.00%
H11	LW=32	12	8.515625	2	20.00%
H11	LW=0.001	12	8.90625	2	20.00%
H12	LW=1	13	34.90625	3	30.00%
H12	LW=32	11	40.75	1	10.00%
H12	LW=0.001	14	40.53125	4	40.00%
H13	LW=1	13	25.875	3	30.00%
H13	LW=32	11	25.671875	1	10.00%
H13	LW=0.001	15	26.5	5	50.00%
H14	LW=1	11	9.984375	1	10.00%
H14	LW=32	11	8.375	1	10.00%
H14	LW=0.001	12	9.328125	2	20.00%
H15	LW=1	11	42.15625	1	10.00%
H15	LW=32	11	34.484375	1	10.00%

H15	LW=0.001	15	41.03125	5	50.00%
H16	LW=1	11	26.640625	1	10.00%
H16	LW=32	11	27.671875	1	10.00%
H16	LW=0.001	15	29.96875	5	50.00%
H17	LW=1	11	9.71875	1	10.00%
H17	LW=32	11	8.28125	1	10.00%
H17	LW=0.001	12	9.46875	2	20.00%
H18	LW=1	11	37.65625	1	10.00%
H18	LW=32	11	34.03125	1	10.00%
H18	LW=0.001	13	41.296875	3	30.00%
H19	LW=1	11	26.953125	1	10.00%
H19	LW=32	11	26.03125	1	10.00%
H19	LW=0.001	15	27.234375	5	50.00%
H20	LW=1	14	9.96875	4	40.00%
H20	LW=32	14	10.0625	4	40.00%
H20	LW=0.001	14	9.953125	4	40.00%
H21	LW=1	13	41.578125	3	30.00%
H21	LW=32	11	41.234375	1	10.00%
H21	LW=0.001	13	47.21875	3	30.00%
H22	LW=1	15	26.234375	5	50.00%
H22	LW=32	11	28.546875	1	10.00%
H22	LW=0.001	15	26.28125	5	50.00%
H23	LW=1	14	9.8125	4	40.00%
H23	LW=32	14	9.65625	4	40.00%
H23	LW=0.001	14	9.875	4	40.00%
H24	LW=1	11	42.5625	1	10.00%
H24	LW=32	11	39.796875	1	10.00%
H24	LW=0.001	13	42.515625	3	30.00%
H25	LW=1	11	28.734375	1	10.00%
H25	LW=32	11	27.4375	1	10.00%
H25	LW=0.001	15	26.609375	5	50.00%
H26	LW=1	14	9.9375	4	40.00%
H26	LW=32	14	10	4	40.00%
H26	LW=0.001	14	9.796875	4	40.00%
H27	LW=1	11	42.8125	1	10.00%
H27	LW=32	11	39.765625	1	10.00%
H27	LW=0.001	13	38.671875	3	30.00%
H28	LW=1	11	30.359375	1	10.00%
H28	LW=32	11	27.6875	1	10.00%
H28	LW=0.001	15	27.875	5	50.00%

Test Instances		T12		
Job Number		4		
Operation Number		57		
Edge Number		47525		
Node Number		580		
Total Length		86		
Average Length		1.50877193		
Optimum Time Slot		31		
LWs	Makespan	Computation Time (s)	Difference	Error Rate
IP	31		0	0.00%
H11 LW=1	38	2501	7	22.58%
H11 LW=86	37	2411.671875	6	19.35%
H11 LW=0.001	38	2687.6875	7	22.58%
H12 LW=1	34	9969.859375	3	9.68%
H12 LW=86	33	8504.421875	2	6.45%
H12 LW=0.001	37	10256.32813	6	19.35%
H13 LW=1	32	8147.390625	1	3.23%
H13 LW=86	32	7004.375	1	3.23%
H13 LW=0.001	32	7272.546875	1	3.23%
H14 LW=1	35	2677.015625	4	12.90%
H14 LW=86	33	2374.625	2	6.45%
H14 LW=0.001	36	2577.234375	5	16.13%
H15 LW=1	33	9168.5	2	6.45%
H15 LW=86	32	7903.546875	1	3.23%
H15 LW=0.001	36	8969.453125	5	16.13%
H16 LW=1	32	6906.96875	1	3.23%
H16 LW=86	32	7000.03125	1	3.23%
H16 LW=0.001	37	7488.453125	6	19.35%
H17 LW=1	33	2150.1875	2	6.45%
H17 LW=86	33	2093.34375	2	6.45%
H17 LW=0.001	35	3252.6875	4	12.90%
H18 LW=1	33	9931.984375	2	6.45%
H18 LW=86	33	7721.515625	2	6.45%
H18 LW=0.001	34	9380.75	3	9.68%
H19 LW=1	31	6177.3125	0	0.00%
H19 LW=86	33	6301.890625	2	6.45%
H19 LW=0.001	35	6288.828125	4	12.90%
H20 LW=1	53	6128.140625	22	70.97%
H20 LW=86	56	5938.296875	25	80.65%
H20 LW=0.001	56	6301.28125	25	80.65%
H21 LW=1	36	20880.75	5	16.13%
H21 LW=86	37	21900.95313	6	19.35%
H21 LW=0.001	40	28674.67188	9	29.03%

H22	LW=1	34	11985.40625	3	9.68%
H22	LW=86	32	9632.203125	1	3.23%
H22	LW=0.001	34	10436.98438	3	9.68%
H23	LW=1	58	4466.6875	27	87.10%
H23	LW=86	58	4620.953125	27	87.10%
H23	LW=0.001	52	5184.75	21	67.74%
H24	LW=1	33	20058.78125	2	6.45%
H24	LW=86	33	21194.48438	2	6.45%
H24	LW=0.001	37	21986.26563	6	19.35%
H25	LW=1	36	9253.5625	5	16.13%
H25	LW=86	35	8798.78125	4	12.90%
H25	LW=0.001	38	9306.609375	7	22.58%
H26	LW=1	58	4614.734375	27	87.10%
H26	LW=86	58	4707.15625	27	87.10%
H26	LW=0.001	58	5331.671875	27	87.10%
H27	LW=1	37	17892.21875	6	19.35%
H27	LW=86	36	19486.51563	5	16.13%
H27	LW=0.001	35	20906.45313	4	12.90%
H28	LW=1	34	8111.1875	3	9.68%
H28	LW=86	32	8122.296875	1	3.23%
H28	LW=0.001	37	9152.96875	6	19.35%

Test Instances		T13			
Job Number		4			
Operation Number		57			
Edge Number		47633			
Node Number		580			
Total Length		86			
Average Length		1.50877193			
Optimum Time Slot		31			
	LWs	Makespan	Computation Time (s)	Difference	Error Rate
IP		31		0	0.00%
H11	LW=1	38	2555.5625	7	22.58%
H11	LW=86	37	2431.875	6	19.35%
H11	LW=0.001	38	3024.703125	7	22.58%
H12	LW=1	32	7894.75	1	3.23%
H12	LW=86	34	7605.375	3	9.68%
H12	LW=0.001	34	8863.046875	3	9.68%
H13	LW=1	32	6516.875	1	3.23%
H13	LW=86	32	6480.6875	1	3.23%

H13	LW=0.001	34	6980.90625	3	9.68%
H14	LW=1	35	3097.09375	4	12.90%
H14	LW=86	33	2909.953125	2	6.45%
H14	LW=0.001	36	4097.328125	5	16.13%
H15	LW=1	34	6263.28125	3	9.68%
H15	LW=86	32	6604.0625	1	3.23%
H15	LW=0.001	37	7961.96875	6	19.35%
H16	LW=1	32	6055.234375	1	3.23%
H16	LW=86	32	6209.828125	1	3.23%
H16	LW=0.001	37	6458.78125	6	19.35%
H17	LW=1	33	2345.40625	2	6.45%
H17	LW=86	33	2359.234375	2	6.45%
H17	LW=0.001	35	2957.5	4	12.90%
H18	LW=1	33	8316.875	2	6.45%
H18	LW=86	32	6929.484375	1	3.23%
H18	LW=0.001	36	8736.875	5	16.13%
H19	LW=1	35	5559.6875	4	12.90%
H19	LW=86	35	5452.921875	4	12.90%
H19	LW=0.001	34	7270.5	3	9.68%
H20	LW=1	53	5720.671875	22	70.97%
H20	LW=86	56	6045.25	25	80.65%
H20	LW=0.001	56	5775.15625	25	80.65%
H21	LW=1	36	21790.65625	5	16.13%
H21	LW=86	37	22983.1875	6	19.35%
H21	LW=0.001	37	21964.4375	6	19.35%
H22	LW=1	35	10497.15625	4	12.90%
H22	LW=86	33	9999.671875	2	6.45%
H22	LW=0.001	34	11175.8125	3	9.68%
H23	LW=1	58	5497.21875	27	87.10%
H23	LW=86	58	5661.234375	27	87.10%
H23	LW=0.001	52	5821.46875	21	67.74%
H24	LW=1	38	23343.67188	7	22.58%
H24	LW=86	34	24148.5625	3	9.68%
H24	LW=0.001	40	23059.95313	9	29.03%
H25	LW=1	34	10292.59375	3	9.68%
H25	LW=86	34	10522.64063	3	9.68%
H25	LW=0.001	37	11597.5625	6	19.35%
H26	LW=1	58	4279.9375	27	87.10%
H26	LW=86	58	4436.140625	27	87.10%
H26	LW=0.001	58	4797.375	27	87.10%
H27	LW=1	36	21057.03125	5	16.13%
H27	LW=86	36	21589.09375	5	16.13%
H27	LW=0.001	40	25291.04688	9	29.03%

H28	LW=1	31	8146	0	0.00%
H28	LW=86	32	8133.625	1	3.23%
H28	LW=0.001	34	7839.28125	3	9.68%

Test Instances		T14			
Job Number		4			
Operation Number		57			
Edge Number		8771			
Node Number		292			
Total Length		86			
Average Length		1.50877193			
Optimum Time Slot		31			
	LWs	Makespan	Computation Time (s)	Difference	Error Rate
IP		31		0	0.00%
H11	LW=1	38	151.34375	7	22.58%
H11	LW=86	37	137.6875	6	19.35%
H11	LW=0.001	38	152.625	7	22.58%
H12	LW=1	35	722.203125	4	12.90%
H12	LW=86	35	665.703125	4	12.90%
H12	LW=0.001	36	726.671875	5	16.13%
H13	LW=1	35	534.671875	4	12.90%
H13	LW=86	33	509.703125	2	6.45%
H13	LW=0.001	35	519.84375	4	12.90%
H14	LW=1	33	130.53125	2	6.45%
H14	LW=86	33	131.625	2	6.45%
H14	LW=0.001	36	142.453125	5	16.13%
H15	LW=1	32	643.828125	1	3.23%
H15	LW=86	32	626.5625	1	3.23%
H15	LW=0.001	36	743.265625	5	16.13%
H16	LW=1	35	456.046875	4	12.90%
H16	LW=86	35	457.515625	4	12.90%
H16	LW=0.001	37	489.53125	6	19.35%
H17	LW=1	34	123.078125	3	9.68%
H17	LW=86	34	122.140625	3	9.68%
H17	LW=0.001	36	129.265625	5	16.13%
H18	LW=1	32	613.40625	1	3.23%
H18	LW=86	32	706.359375	1	3.23%
H18	LW=0.001	33	740.578125	2	6.45%
H19	LW=1	34	431.609375	3	9.68%
H19	LW=86	34	449.515625	3	9.68%

H19	LW=0.001	34	466.421875	3	9.68%
H20	LW=1	43	286.25	12	38.71%
H20	LW=86	43	284.109375	12	38.71%
H20	LW=0.001	43	293.3125	12	38.71%
H21	LW=1	37	1727.375	6	19.35%
H21	LW=86	37	1675.9375	6	19.35%
H21	LW=0.001	35	1484.4375	4	12.90%
H22	LW=1	34	600.234375	3	9.68%
H22	LW=86	32	582.46875	1	3.23%
H22	LW=0.001	36	559.984375	5	16.13%
H23	LW=1	43	287.78125	12	38.71%
H23	LW=86	43	294.59375	12	38.71%
H23	LW=0.001	43	299.53125	12	38.71%
H24	LW=1	36	1776.796875	5	16.13%
H24	LW=86	36	1734.140625	5	16.13%
H24	LW=0.001	35	1504.28125	4	12.90%
H25	LW=1	31	670.125	0	0.00%
H25	LW=86	35	562.609375	4	12.90%
H25	LW=0.001	37	542.125	6	19.35%
H26	LW=1	43	273.828125	12	38.71%
H26	LW=86	43	281.421875	12	38.71%
H26	LW=0.001	43	294.859375	12	38.71%
H27	LW=1	36	1833.34375	5	16.13%
H27	LW=86	36	1840.421875	5	16.13%
H27	LW=0.001	36	1746.0625	5	16.13%
H28	LW=1	33	599.5	2	6.45%
H28	LW=86	33	604.6875	2	6.45%
H28	LW=0.001	36	590.28125	5	16.13%

Test Instances		T15			
Job Number		4			
Operation Number		13			
Edge Number		989			
Node Number		60			
Total Length		24			
Average Length		1.846153846			
Optimum Time Slot		17			
LWs		Makespan	Computation Time (s)	Difference	Error Rate
IP		17	989.2135717	0	0.00%
H1	LW=1	17	6.828125	0	0.00%

H1	LW=24	17	7.125	0	0.00%
H1	LW=0.001	19	7.8125	2	11.76%
H2	LW=1	17	6.4375	0	0.00%
H2	LW=24	17	6.1875	0	0.00%
H2	LW=0.001	17	6.40625	0	0.00%
H3	LW=1	17	6.40625	0	0.00%
H3	LW=24	17	6.53125	0	0.00%
H3	LW=0.001	17	6.78125	0	0.00%
H4	LW=1	18	6.59375	1	5.88%
H4	LW=24	18	5.609375	1	5.88%
H4	LW=0.001	19	6.328125	2	11.76%
H5	LW=1	18	6.265625	1	5.88%
H5	LW=24	18	5.703125	1	5.88%
H5	LW=0.001	17	6.34375	0	0.00%
H6	LW=1	17	5.28125	0	0.00%
H6	LW=24	17	5.90625	0	0.00%
H6	LW=0.001	17	5.59375	0	0.00%
H7	LW=1	17	6.59375	0	0.00%
H7	LW=24	17	6.671875	0	0.00%
H7	LW=0.001	17	6.53125	0	0.00%
H8	LW=1	18	5.953125	1	5.88%
H8	LW=24	18	5.8125	1	5.88%
H8	LW=0.001	18	6.0625	1	5.88%
H9	LW=1	17	5.640625	0	0.00%
H9	LW=24	17	5.515625	0	0.00%
H9	LW=0.001	17	5.15625	0	0.00%
H10	LW=1	17	5.421875	0	0.00%
H10	LW=24	17	5.4375	0	0.00%
H10	LW=0.001	17	5.484375	0	0.00%
H11	LW=1	17	0.5	0	0.00%
H11	LW=24	17	0.5	0	0.00%
H11	LW=0.001	17	0.515625	0	0.00%
H12	LW=1	17	2.890625	0	0.00%
H12	LW=24	17	2.890625	0	0.00%
H12	LW=0.001	19	3.421875	2	11.76%
H13	LW=1	17	2.296875	0	0.00%
H13	LW=24	17	2.3125	0	0.00%
H13	LW=0.001	19	2.9375	2	11.76%
H14	LW=1	17	0.5	0	0.00%
H14	LW=24	17	0.5	0	0.00%
H14	LW=0.001	17	0.5	0	0.00%
H15	LW=1	17	2.734375	0	0.00%
H15	LW=24	17	2.84375	0	0.00%

H15	LW=0.001	17	2.65625	0	0.00%
H16	LW=1	17	2.1875	0	0.00%
H16	LW=24	17	2.15625	0	0.00%
H16	LW=0.001	17	2.1875	0	0.00%
H17	LW=1	17	0.5	0	0.00%
H17	LW=24	17	0.5	0	0.00%
H17	LW=0.001	17	0.453125	0	0.00%
H18	LW=1	17	2.9375	0	0.00%
H18	LW=24	17	2.84375	0	0.00%
H18	LW=0.001	17	3.421875	0	0.00%
H19	LW=1	17	2.421875	0	0.00%
H19	LW=24	17	2.390625	0	0.00%
H19	LW=0.001	17	2.25	0	0.00%
H20	LW=1	18	0.515625	1	5.88%
H20	LW=24	18	0.515625	1	5.88%
H20	LW=0.001	18	0.5625	1	5.88%
H21	LW=1	17	2.90625	0	0.00%
H21	LW=24	17	2.90625	0	0.00%
H21	LW=0.001	17	2.8125	0	0.00%
H22	LW=1	17	2.390625	0	0.00%
H22	LW=24	17	2.328125	0	0.00%
H22	LW=0.001	19	3.109375	2	11.76%
H23	LW=1	18	0.53125	1	5.88%
H23	LW=24	18	0.5625	1	5.88%
H23	LW=0.001	18	0.53125	1	5.88%
H24	LW=1	17	3.125	0	0.00%
H24	LW=24	17	3.125	0	0.00%
H24	LW=0.001	17	2.875	0	0.00%
H25	LW=1	17	2.375	0	0.00%
H25	LW=24	17	2.3125	0	0.00%
H25	LW=0.001	17	2.359375	0	0.00%
H26	LW=1	18	0.5	1	5.88%
H26	LW=24	18	0.5	1	5.88%
H26	LW=0.001	18	0.578125	1	5.88%
H27	LW=1	17	3.046875	0	0.00%
H27	LW=24	17	3.015625	0	0.00%
H27	LW=0.001	17	3.75	0	0.00%
H28	LW=1	17	2.34375	0	0.00%
H28	LW=24	17	2.359375	0	0.00%
H28	LW=0.001	17	2.40625	0	0.00%

Test Instances		T16		
Job Number		5		
Operation Number		15		
Edge Number		689		
Node Number		51		
Total Length		21.5		
Average Length		1.433333333		
Optimum Time Slot		10		
LWs	Makespan	Computation Time (s)	Difference	Error Rate
IP	10		0	0.00%
H1 LW=1	11	5.90625	1	10.00%
H1 LW=21.5	10	6.015625	0	0.00%
H1 LW=0.001	11	5.875	1	10.00%
H2 LW=1	12	5.125	2	20.00%
H2 LW=21.5	10	5.40625	0	0.00%
H2 LW=0.001	12	5.375	2	20.00%
H3 LW=1	10	5.421875	0	0.00%
H3 LW=21.5	10	5.5625	0	0.00%
H3 LW=0.001	11	5.625	1	10.00%
H4 LW=1	10	5.5	0	0.00%
H4 LW=21.5	10	5.5	0	0.00%
H4 LW=0.001	11	5.40625	1	10.00%
H5 LW=1	12	4.953125	2	20.00%
H5 LW=21.5	10	5.3125	0	0.00%
H5 LW=0.001	13	5.796875	3	30.00%
H6 LW=1	12	5.625	2	20.00%
H6 LW=21.5	11	4.734375	1	10.00%
H6 LW=0.001	14	5.5	4	40.00%
H7 LW=1	12	5.875	2	20.00%
H7 LW=21.5	12	5.921875	2	20.00%
H7 LW=0.001	12	6.125	2	20.00%
H8 LW=1	10	5.46875	0	0.00%
H8 LW=21.5	10	5.4375	0	0.00%
H8 LW=0.001	11	5.296875	1	10.00%
H9 LW=1	11	4.765625	1	10.00%
H9 LW=21.5	11	4.890625	1	10.00%
H9 LW=0.001	13	5.640625	3	30.00%
H10 LW=1	11	4.84375	1	10.00%
H10 LW=21.5	11	4.625	1	10.00%
H10 LW=0.001	13	5.453125	3	30.00%
H11 LW=1	11	0.234375	1	10.00%
H11 LW=21.5	12	0.25	2	20.00%
H11 LW=0.001	11	0.265625	1	10.00%

H12	LW=1	12	1.4375	2	20.00%
H12	LW=21.5	11	1.953125	1	10.00%
H12	LW=0.001	12	1.546875	2	20.00%
H13	LW=1	11	1.15625	1	10.00%
H13	LW=21.5	10	1.15625	0	0.00%
H13	LW=0.001	11	1.21875	1	10.00%
H14	LW=1	12	0.265625	2	20.00%
H14	LW=21.5	12	0.234375	2	20.00%
H14	LW=0.001	14	0.25	4	40.00%
H15	LW=1	12	1.953125	2	20.00%
H15	LW=21.5	11	1.515625	1	10.00%
H15	LW=0.001	13	1.765625	3	30.00%
H16	LW=1	10	1.125	0	0.00%
H16	LW=21.5	10	1.15625	0	0.00%
H16	LW=0.001	13	1.3125	3	30.00%
H17	LW=1	12	0.28125	2	20.00%
H17	LW=21.5	12	0.265625	2	20.00%
H17	LW=0.001	11	0.25	1	10.00%
H18	LW=1	11	1.609375	1	10.00%
H18	LW=21.5	11	1.53125	1	10.00%
H18	LW=0.001	11	2.0625	1	10.00%
H19	LW=1	10	1.125	0	0.00%
H19	LW=21.5	10	1.171875	0	0.00%
H19	LW=0.001	11	1.234375	1	10.00%
H20	LW=1	14	0.265625	4	40.00%
H20	LW=21.5	14	0.234375	4	40.00%
H20	LW=0.001	14	0.234375	4	40.00%
H21	LW=1	12	1.21875	2	20.00%
H21	LW=21.5	11	1.234375	1	10.00%
H21	LW=0.001	12	1.34375	2	20.00%
H22	LW=1	12	1.15625	2	20.00%
H22	LW=21.5	11	1.140625	1	10.00%
H22	LW=0.001	12	1.125	2	20.00%
H23	LW=1	14	0.25	4	40.00%
H23	LW=21.5	14	0.25	4	40.00%
H23	LW=0.001	14	0.25	4	40.00%
H24	LW=1	12	1.34375	2	20.00%
H24	LW=21.5	11	1.203125	1	10.00%
H24	LW=0.001	12	1.46875	2	20.00%
H25	LW=1	12	1.046875	2	20.00%
H25	LW=21.5	11	1	1	10.00%
H25	LW=0.001	12	1.078125	2	20.00%
H26	LW=1	14	0.234375	4	40.00%

H26	LW=21.5	14	0.25	4	40.00%
H26	LW=0.001	14	0.234375	4	40.00%
H27	LW=1	12	1.25	2	20.00%
H27	LW=21.5	11	1.1875	1	10.00%
H27	LW=0.001	12	1.296875	2	20.00%
H28	LW=1	12	1.125	2	20.00%
H28	LW=21.5	11	1.109375	1	10.00%
H28	LW=0.001	12	1.140625	2	20.00%

Test Instances			T17		
Job Number			4		
Operation Number			14		
Edge Number			2184		
Node Number			110		
Total Length			31.5		
Average Length			2.25		
Optimum Time Slot			10		
	LWs	Makespan	Computation Time (s)	Difference	Error Rate
IP		10		0	0.00%
H1	LW=1	12	500.90625	2	20.00%
H1	LW=31.5	11	495.59375	1	10.00%
H1	LW=0.001	16	528.09375	6	60.00%
H2	LW=1	11	497	1	10.00%
H2	LW=31.5	12	486.21875	2	20.00%
H2	LW=0.001	16	521.765625	6	60.00%
H3	LW=1	12	510.8125	2	20.00%
H3	LW=31.5	12	514.25	2	20.00%
H3	LW=0.001	16	530.171875	6	60.00%
H4	LW=1	12	458.328125	2	20.00%
H4	LW=31.5	11	455.265625	1	10.00%
H4	LW=0.001	16	493.21875	6	60.00%
H5	LW=1	11	439.5	1	10.00%
H5	LW=31.5	11	452.453125	1	10.00%
H5	LW=0.001	16	479.78125	6	60.00%
H6	LW=1	12	454.046875	2	20.00%
H6	LW=31.5	11	450.9375	1	10.00%
H6	LW=0.001	16	487.25	6	60.00%
H7	LW=1	12	456.703125	2	20.00%

H7	LW=31.5	12	475.203125	2	20.00%
H7	LW=0.001	16	504.796875	6	60.00%
H8	LW=1	11	451.640625	1	10.00%
H8	LW=31.5	11	457.84375	1	10.00%
H8	LW=0.001	11	494.109375	1	10.00%
H9	LW=1	11	455.59375	1	10.00%
H9	LW=31.5	11	461.234375	1	10.00%
H9	LW=0.001	16	512.53125	6	60.00%
H10	LW=1	14	492.71875	4	40.00%
H10	LW=31.5	11	452.484375	1	10.00%
H10	LW=0.001	16	491.03125	6	60.00%
H11	LW=1	13	2.765625	3	30.00%
H11	LW=31.5	10	1.796875	0	0.00%
H11	LW=0.001	13	2.359375	3	30.00%
H12	LW=1	13	12.140625	3	30.00%
H12	LW=31.5	12	9.765625	2	20.00%
H12	LW=0.001	13	9.171875	3	30.00%
H13	LW=1	12	6.46875	2	20.00%
H13	LW=31.5	12	6.734375	2	20.00%
H13	LW=0.001	13	7.546875	3	30.00%
H14	LW=1	13	2.21875	3	30.00%
H14	LW=31.5	10	1.84375	0	0.00%
H14	LW=0.001	13	2.234375	3	30.00%
H15	LW=1	12	9.921875	2	20.00%
H15	LW=31.5	11	8.015625	1	10.00%
H15	LW=0.001	13	11.234375	3	30.00%
H16	LW=1	12	6.359375	2	20.00%
H16	LW=31.5	11	6.375	1	10.00%
H16	LW=0.001	13	7.4375	3	30.00%
H17	LW=1	11	1.9375	1	10.00%
H17	LW=31.5	10	1.8125	0	0.00%
H17	LW=0.001	12	1.9375	2	20.00%
H18	LW=1	11	9.515625	1	10.00%
H18	LW=31.5	11	9.15625	1	10.00%
H18	LW=0.001	13	8.765625	3	30.00%
H19	LW=1	12	7.0625	2	20.00%
H19	LW=31.5	12	6.609375	2	20.00%
H19	LW=0.001	12	6.890625	2	20.00%
H20	LW=1	12	2.421875	2	20.00%
H20	LW=31.5	12	2.421875	2	20.00%
H20	LW=0.001	12	2.515625	2	20.00%
H21	LW=1	14	10.796875	4	40.00%
H21	LW=31.5	12	10.359375	2	20.00%

H21	LW=0.001	14	10.734375	4	40.00%
H22	LW=1	15	7.03125	5	50.00%
H22	LW=31.5	11	6.1875	1	10.00%
H22	LW=0.001	15	6.84375	5	50.00%
H23	LW=1	12	2.484375	2	20.00%
H23	LW=31.5	12	2.515625	2	20.00%
H23	LW=0.001	12	2.4375	2	20.00%
H24	LW=1	12	9.4375	2	20.00%
H24	LW=31.5	12	9.59375	2	20.00%
H24	LW=0.001	14	10.546875	4	40.00%
H25	LW=1	11	6.5625	1	10.00%
H25	LW=31.5	12	6.78125	2	20.00%
H25	LW=0.001	15	6.984375	5	50.00%
H26	LW=1	12	2.546875	2	20.00%
H26	LW=31.5	12	2.453125	2	20.00%
H26	LW=0.001	12	2.421875	2	20.00%
H27	LW=1	12	9.875	2	20.00%
H27	LW=31.5	12	9.125	2	20.00%
H27	LW=0.001	12	9.125	2	20.00%
H28	LW=1	12	6.734375	2	20.00%
H28	LW=31.5	12	12	2	20.00%
H28	LW=0.001	15	7.09375	5	50.00%

Test Instances			T18		
Job Number			4		
Operation Number			14		
Edge Number			3108		
Node Number			126		
Total Length			31.5		
Average Length			2.25		
Optimum Time Slot			10		
	LWs	Makespan	Computation Time (s)	Difference	Error Rate
IP		10		0	0.00%
H1	LW=1	13	1354.8125	3	30.00%
H1	LW=31.5	11	1413.015625	1	10.00%
H1	LW=0.001	16	1339.40625	6	60.00%
H2	LW=1	11	1036.546875	1	10.00%
H2	LW=31.5	11	1047.0625	1	10.00%
H2	LW=0.001	15	1047.59375	5	50.00%
H3	LW=1	11	988.296875	1	10.00%

H3	LW=31.5	11	1021.28125	1	10.00%
H3	LW=0.001	15	1027.6875	5	50.00%
H4	LW=1	13	939.53125	3	30.00%
H4	LW=31.5	11	946.796875	1	10.00%
H4	LW=0.001	16	940.5	6	60.00%
H5	LW=1	11	989.84375	1	10.00%
H5	LW=31.5	11	939.4375	1	10.00%
H5	LW=0.001	16	936.671875	6	60.00%
H6	LW=1	13	955.3125	3	30.00%
H6	LW=31.5	11	967.953125	1	10.00%
H6	LW=0.001	16	1024.765625	6	60.00%
H7	LW=1	13	1025.21875	3	30.00%
H7	LW=31.5	13	1012.78125	3	30.00%
H7	LW=0.001	16	1023.578125	6	60.00%
H8	LW=1	11	971.671875	1	10.00%
H8	LW=31.5	11	976.546875	1	10.00%
H8	LW=0.001	15	975.328125	5	50.00%
H9	LW=1	11	996.578125	1	10.00%
H9	LW=31.5	11	990.40625	1	10.00%
H9	LW=0.001	15	987.546875	5	50.00%
H10	LW=1	14	1006.203125	4	40.00%
H10	LW=31.5	11	1005.4375	1	10.00%
H10	LW=0.001	16	1018.65625	6	60.00%
H11	LW=1	13	4.15625	3	30.00%
H11	LW=31.5	10	3.671875	0	0.00%
H11	LW=0.001	15	4.1875	5	50.00%
H12	LW=1	13	17.5625	3	30.00%
H12	LW=31.5	10	14.8125	0	0.00%
H12	LW=0.001	14	14.265625	4	40.00%
H13	LW=1	13	11.59375	3	30.00%
H13	LW=31.5	11	11.65625	1	10.00%
H13	LW=0.001	14	11.75	4	40.00%
H14	LW=1	11	4.390625	1	10.00%
H14	LW=31.5	10	3.609375	0	0.00%
H14	LW=0.001	13	4.234375	3	30.00%
H15	LW=1	11	21.71875	1	10.00%
H15	LW=31.5	10	20.484375	0	0.00%
H15	LW=0.001	13	14.65625	3	30.00%
H16	LW=1	13	12.1875	3	30.00%
H16	LW=31.5	11	11.515625	1	10.00%
H16	LW=0.001	14	11.453125	4	40.00%
H17	LW=1	11	4.15625	1	10.00%
H17	LW=31.5	11	4.046875	1	10.00%

H17	LW=0.001	14	3.640625	4	40.00%
H18	LW=1	10	14.109375	0	0.00%
H18	LW=31.5	10	15.640625	0	0.00%
H18	LW=0.001	13	15.78125	3	30.00%
H19	LW=1	11	11.25	1	10.00%
H19	LW=31.5	11	10.984375	1	10.00%
H19	LW=0.001	16	12.890625	6	60.00%
H20	LW=1	14	5.796875	4	40.00%
H20	LW=31.5	14	5.8125	4	40.00%
H20	LW=0.001	14	5.8125	4	40.00%
H21	LW=1	11	15.84375	1	10.00%
H21	LW=31.5	11	16.25	1	10.00%
H21	LW=0.001	11	16.140625	1	10.00%
H22	LW=1	15	11.40625	5	50.00%
H22	LW=31.5	12	12.46875	2	20.00%
H22	LW=0.001	15	11.53125	5	50.00%
H23	LW=1	14	5.8125	4	40.00%
H23	LW=31.5	14	6.125	4	40.00%
H23	LW=0.001	14	5.890625	4	40.00%
H24	LW=1	11	15.890625	1	10.00%
H24	LW=31.5	11	15.921875	1	10.00%
H24	LW=0.001	11	16.1875	1	10.00%
H25	LW=1	13	11.234375	3	30.00%
H25	LW=31.5	11	10.828125	1	10.00%
H25	LW=0.001	15	11.296875	5	50.00%
H26	LW=1	14	5.9375	4	40.00%
H26	LW=31.5	14	5.9375	4	40.00%
H26	LW=0.001	14	5.875	4	40.00%
H27	LW=1	14	16.40625	4	40.00%
H27	LW=31.5	14	16.625	4	40.00%
H27	LW=0.001	14	16.625	4	40.00%
H28	LW=1	12	11.34375	2	20.00%
H28	LW=31.5	14	11.71875	4	40.00%
H28	LW=0.001	15	11.21875	5	50.00%

Test Instances	T19
Job Number	4
Operation Number	14
Edge Number	1387
Node Number	89

Total Length		30			
Average Length		2.142857143			
Optimum Time Slot		10			
	LWs	Makespan	Computation Time (s)	Difference	Error Rate
IP		10		0	0.00%
H1	LW=1	13	109.265625	3	30.00%
H1	LW=30	11	108.609375	1	10.00%
H1	LW=0.001	15	119.40625	5	50.00%
H2	LW=1	12	107.890625	2	20.00%
H2	LW=30	11	105.5	1	10.00%
H2	LW=0.001	15	119.21875	5	50.00%
H3	LW=1	11	112.71875	1	10.00%
H3	LW=30	11	107.65625	1	10.00%
H3	LW=0.001	13	115.390625	3	30.00%
H4	LW=1	12	110.203125	2	20.00%
H4	LW=30	11	105.203125	1	10.00%
H4	LW=0.001	15	114.421875	5	50.00%
H5	LW=1	13	114.375	3	30.00%
H5	LW=30	11	108.703125	1	10.00%
H5	LW=0.001	15	116.0625	5	50.00%
H6	LW=1	12	115.328125	2	20.00%
H6	LW=30	11	112.5625	1	10.00%
H6	LW=0.001	15	118.328125	5	50.00%
H7	LW=1	12	121.015625	2	20.00%
H7	LW=30	11	114.25	1	10.00%
H7	LW=0.001	13	115.828125	3	30.00%
H8	LW=1	11	110.125	1	10.00%
H8	LW=30	11	106.734375	1	10.00%
H8	LW=0.001	15	115.671875	5	50.00%
H9	LW=1	11	110.15625	1	10.00%
H9	LW=30	11	111.96875	1	10.00%
H9	LW=0.001	15	118.8125	5	50.00%
H10	LW=1	12	114.46875	2	20.00%
H10	LW=30	11	112.515625	1	10.00%
H10	LW=0.001	15	119.59375	5	50.00%
H11	LW=1	11	1.078125	1	10.00%
H11	LW=30	11	1.09375	1	10.00%
H11	LW=0.001	15	1.125	5	50.00%
H12	LW=1	11	4.9375	1	10.00%
H12	LW=30	11	5.234375	1	10.00%
H12	LW=0.001	12	4.5	2	20.00%
H13	LW=1	13	3.96875	3	30.00%
H13	LW=30	11	3.65625	1	10.00%

H13	LW=0.001	12	4.046875	2	20.00%
H14	LW=1	12	1.125	2	20.00%
H14	LW=30	11	1.03125	1	10.00%
H14	LW=0.001	15	1.125	5	50.00%
H15	LW=1	12	4.71875	2	20.00%
H15	LW=30	11	4.703125	1	10.00%
H15	LW=0.001	13	4.9375	3	30.00%
H16	LW=1	12	3.46875	2	20.00%
H16	LW=30	11	3.78125	1	10.00%
H16	LW=0.001	13	3.96875	3	30.00%
H17	LW=1	12	1.046875	2	20.00%
H17	LW=30	11	1.078125	1	10.00%
H17	LW=0.001	11	0.984375	1	10.00%
H18	LW=1	12	4.796875	2	20.00%
H18	LW=30	11	4.984375	1	10.00%
H18	LW=0.001	12	4.75	2	20.00%
H19	LW=1	11	3.6875	1	10.00%
H19	LW=30	11	3.609375	1	10.00%
H19	LW=0.001	15	4.21875	5	50.00%
H20	LW=1	14	1.203125	4	40.00%
H20	LW=30	14	1.296875	4	40.00%
H20	LW=0.001	14	1.21875	4	40.00%
H21	LW=1	11	5.40625	1	10.00%
H21	LW=30	11	4.6875	1	10.00%
H21	LW=0.001	13	4.53125	3	30.00%
H22	LW=1	12	3.4375	2	20.00%
H22	LW=30	11	3.640625	1	10.00%
H22	LW=0.001	12	3.375	2	20.00%
H23	LW=1	14	1.21875	4	40.00%
H23	LW=30	14	1.203125	4	40.00%
H23	LW=0.001	14	1.171875	4	40.00%
H24	LW=1	11	5.28125	1	10.00%
H24	LW=30	11	4.296875	1	10.00%
H24	LW=0.001	13	4.015625	3	30.00%
H25	LW=1	12	3.359375	2	20.00%
H25	LW=30	11	3.390625	1	10.00%
H25	LW=0.001	12	3.203125	2	20.00%
H26	LW=1	14	1.15625	4	40.00%
H26	LW=30	14	1.28125	4	40.00%
H26	LW=0.001	14	1.296875	4	40.00%
H27	LW=1	11	4.5	1	10.00%
H27	LW=30	11	4.65625	1	10.00%
H27	LW=0.001	13	4.453125	3	30.00%

H28	LW=1	11	3.5	1	10.00%
H28	LW=30	11	3.640625	1	10.00%
H28	LW=0.001	12	3.390625	2	20.00%

Test Instances		T20			
Job Number		4			
Operation Number		11			
Edge Number		387			
Node Number		41			
Total Length		17.5			
Average Length		1.590909091			
Optimum Time Slot		8			
	LWs	Makespan	Computation Time (s)	Difference	Error Rate
IP		8	241.4488821	0	0.00%
H1	LW=1	9	1.796875	1	12.50%
H1	LW=17.5	9	1.78125	1	12.50%
H1	LW=0.001	10	1.953125	2	25.00%
H2	LW=1	9	1.921875	1	12.50%
H2	LW=17.5	9	1.6875	1	12.50%
H2	LW=0.001	10	1.8125	2	25.00%
H3	LW=1	9	1.6875	1	12.50%
H3	LW=17.5	9	1.703125	1	12.50%
H3	LW=0.001	9	1.96875	1	12.50%
H4	LW=1	9	1.78125	1	12.50%
H4	LW=17.5	9	1.640625	1	12.50%
H4	LW=0.001	10	1.765625	2	25.00%
H5	LW=1	9	1.875	1	12.50%
H5	LW=17.5	9	1.828125	1	12.50%
H5	LW=0.001	10	2.171875	2	25.00%
H6	LW=1	9	1.859375	1	12.50%
H6	LW=17.5	9	1.609375	1	12.50%
H6	LW=0.001	10	2.1875	2	25.00%
H7	LW=1	10	2.046875	2	25.00%
H7	LW=17.5	10	2.078125	2	25.00%
H7	LW=0.001	10	2.046875	2	25.00%
H8	LW=1	9	1.609375	1	12.50%
H8	LW=17.5	9	1.609375	1	12.50%
H8	LW=0.001	9	1.890625	1	12.50%
H9	LW=1	9	1.625	1	12.50%
H9	LW=17.5	9	1.625	1	12.50%

H9	LW=0.001	9	1.625	1	12.50%
H10	LW=1	9	1.625	1	12.50%
H10	LW=17.5	9	1.59375	1	12.50%
H10	LW=0.001	11	2.265625	3	37.50%
H11	LW=1	10	0.09375	2	25.00%
H11	LW=17.5	9	0.078125	1	12.50%
H11	LW=0.001	13	0.09375	5	62.50%
H12	LW=1	10	0.8125	2	25.00%
H12	LW=17.5	9	0.65625	1	12.50%
H12	LW=0.001	12	1.046875	4	50.00%
H13	LW=1	9	0.390625	1	12.50%
H13	LW=17.5	9	0.40625	1	12.50%
H13	LW=0.001	10	0.4375	2	25.00%
H14	LW=1	8	0.09375	0	0.00%
H14	LW=17.5	9	0.09375	1	12.50%
H14	LW=0.001	10	0.09375	2	25.00%
H15	LW=1	9	0.65625	1	12.50%
H15	LW=17.5	9	0.640625	1	12.50%
H15	LW=0.001	10	0.75	2	25.00%
H16	LW=1	9	0.46875	1	12.50%
H16	LW=17.5	9	0.390625	1	12.50%
H16	LW=0.001	10	0.4375	2	25.00%
H17	LW=1	8	0.09375	0	0.00%
H17	LW=17.5	9	0.09375	1	12.50%
H17	LW=0.001	8	0.09375	0	0.00%
H18	LW=1	9	0.609375	1	12.50%
H18	LW=17.5	9	0.578125	1	12.50%
H18	LW=0.001	9	0.703125	1	12.50%
H19	LW=1	9	0.40625	1	12.50%
H19	LW=17.5	9	0.40625	1	12.50%
H19	LW=0.001	9	0.46875	1	12.50%
H20	LW=1	10	0.09375	2	25.00%
H20	LW=17.5	11	0.109375	3	37.50%
H20	LW=0.001	10	0.09375	2	25.00%
H21	LW=1	9	0.59375	1	12.50%
H21	LW=17.5	9	0.46875	1	12.50%
H21	LW=0.001	10	0.546875	2	25.00%
H22	LW=1	9	0.375	1	12.50%
H22	LW=17.5	9	0.375	1	12.50%
H22	LW=0.001	10	0.421875	2	25.00%
H23	LW=1	10	0.09375	2	25.00%
H23	LW=17.5	11	0.09375	3	37.50%
H23	LW=0.001	10	0.109375	2	25.00%

H24	LW=1	9	0.53125	1	12.50%
H24	LW=17.5	9	0.453125	1	12.50%
H24	LW=0.001	8	0.515625	0	0.00%
H25	LW=1	9	0.4375	1	12.50%
H25	LW=17.5	9	0.375	1	12.50%
H25	LW=0.001	10	0.4375	2	25.00%
H26	LW=1	11	0.109375	3	37.50%
H26	LW=17.5	11	0.09375	3	37.50%
H26	LW=0.001	10	0.109375	2	25.00%
H27	LW=1	9	0.484375	1	12.50%
H27	LW=17.5	9	0.4375	1	12.50%
H27	LW=0.001	9	0.546875	1	12.50%
H28	LW=1	9	0.390625	1	12.50%
H28	LW=17.5	9	0.390625	1	12.50%
H28	LW=0.001	9	0.46875	1	12.50%

Test Instances			T21		
Job Number			5		
Operation Number			12		
Edge Number			583		
Node Number			47		
Total Length			19.5		
Average Length			1.625		
Optimum Time Slot			9		
	LWs	Makespan	Computation Time (s)	Difference	Error Rate
IP		9	1903.757093	0	0.00%
H1	LW=1	9	3.796875	0	0.00%
H1	LW=19.5	9	3.796875	0	0.00%
H1	LW=0.001	12	4	3	33.33%
H2	LW=1	10	4.0625	1	11.11%
H2	LW=19.5	9	3.609375	0	0.00%
H2	LW=0.001	12	3.8125	3	33.33%
H3	LW=1	9	3.671875	0	0.00%
H3	LW=19.5	9	3.53125	0	0.00%
H3	LW=0.001	10	4.109375	1	11.11%
H4	LW=1	9	3.515625	0	0.00%
H4	LW=19.5	9	3.546875	0	0.00%
H4	LW=0.001	12	3.59375	3	33.33%
H5	LW=1	11	4	2	22.22%
H5	LW=19.5	9	3.578125	0	0.00%

H5	LW=0.001	12	4.15625	3	33.33%
H6	LW=1	12	4.015625	3	33.33%
H6	LW=19.5	9	3.46875	0	0.00%
H6	LW=0.001	12	4.1875	3	33.33%
H7	LW=1	12	4.46875	3	33.33%
H7	LW=19.5	12	4.40625	3	33.33%
H7	LW=0.001	12	4.171875	3	33.33%
H8	LW=1	9	3.484375	0	0.00%
H8	LW=19.5	9	3.484375	0	0.00%
H8	LW=0.001	10	4.46875	1	11.11%
H9	LW=1	9	3.515625	0	0.00%
H9	LW=19.5	9	3.515625	0	0.00%
H9	LW=0.001	10	4.671875	1	11.11%
H10	LW=1	9	3.484375	0	0.00%
H10	LW=19.5	9	3.515625	0	0.00%
H10	LW=0.001	10	3.84375	1	11.11%
H11	LW=1	10	0.171875	1	11.11%
H11	LW=19.5	9	0.171875	0	0.00%
H11	LW=0.001	15	0.1875	6	66.67%
H12	LW=1	10	1.6875	1	11.11%
H12	LW=19.5	9	1.203125	0	0.00%
H12	LW=0.001	14	1.6875	5	55.56%
H13	LW=1	9	0.671875	0	0.00%
H13	LW=19.5	9	0.6875	0	0.00%
H13	LW=0.001	12	0.78125	3	33.33%
H14	LW=1	10	0.15625	1	11.11%
H14	LW=19.5	9	0.171875	0	0.00%
H14	LW=0.001	12	0.171875	3	33.33%
H15	LW=1	9	1.171875	0	0.00%
H15	LW=19.5	9	0.953125	0	0.00%
H15	LW=0.001	12	1.203125	3	33.33%
H16	LW=1	10	0.796875	1	11.11%
H16	LW=19.5	9	0.6875	0	0.00%
H16	LW=0.001	12	0.84375	3	33.33%
H17	LW=1	10	0.171875	1	11.11%
H17	LW=19.5	9	0.171875	0	0.00%
H17	LW=0.001	10	0.1875	1	11.11%
H18	LW=1	9	1.046875	0	0.00%
H18	LW=19.5	9	1.203125	0	0.00%
H18	LW=0.001	9	1.09375	0	0.00%
H19	LW=1	9	0.65625	0	0.00%
H19	LW=19.5	9	0.75	0	0.00%
H19	LW=0.001	10	0.78125	1	11.11%

H20	LW=1	11	0.1875	2	22.22%
H20	LW=19.5	13	0.203125	4	44.44%
H20	LW=0.001	11	0.265625	2	22.22%
H21	LW=1	9	0.859375	0	0.00%
H21	LW=19.5	9	1.03125	0	0.00%
H21	LW=0.001	12	0.859375	3	33.33%
H22	LW=1	9	0.71875	0	0.00%
H22	LW=19.5	9	0.6875	0	0.00%
H22	LW=0.001	12	0.765625	3	33.33%
H23	LW=1	11	0.234375	2	22.22%
H23	LW=19.5	19	0.203125	10	111.11%
H23	LW=0.001	11	0.1875	2	22.22%
H24	LW=1	10	1.171875	1	11.11%
H24	LW=19.5	9	0.953125	0	0.00%
H24	LW=0.001	12	1.171875	3	33.33%
H25	LW=1	10	0.765625	1	11.11%
H25	LW=19.5	10	0.8125	1	11.11%
H25	LW=0.001	12	0.90625	3	33.33%
H26	LW=1	11	0.1875	2	22.22%
H26	LW=19.5	13	0.203125	4	44.44%
H26	LW=0.001	11	0.203125	2	22.22%
H27	LW=1	9	0.90625	0	0.00%
H27	LW=19.5	9	1	0	0.00%
H27	LW=0.001	10	1.171875	1	11.11%
H28	LW=1	9	0.671875	0	0.00%
H28	LW=19.5	9	0.671875	0	0.00%
H28	LW=0.001	10	0.75	1	11.11%

Test Instances			T22		
Job Number			3		
Operation Number			10		
Edge Number			247		
Node Number			39		
Total Length			19.5		
Average Length			1.95		
Optimum Time Slot			10		
	LWs	Makespan	Computation Time (s)	Difference	Error Rate
IP		10	61.10258196	0	0.00%
H1	LW=1	13	1.28125	3	30.00%
H1	LW=19.5	13	1.296875	3	30.00%

H1	LW=0.001	13	1.28125	3	30.00%
H2	LW=1	13	1.25	3	30.00%
H2	LW=19.5	13	1.21875	3	30.00%
H2	LW=0.001	10	1.265625	0	0.00%
H3	LW=1	10	1.296875	0	0.00%
H3	LW=19.5	13	1.296875	3	30.00%
H3	LW=0.001	11	1.3125	1	10.00%
H4	LW=1	11	1.4375	1	10.00%
H4	LW=19.5	11	1.421875	1	10.00%
H4	LW=0.001	13	1.1875	3	30.00%
H5	LW=1	10	1.40625	0	0.00%
H5	LW=19.5	11	1.4375	1	10.00%
H5	LW=0.001	11	1.4375	1	10.00%
H6	LW=1	10	1.453125	0	0.00%
H6	LW=19.5	11	1.4375	1	10.00%
H6	LW=0.001	11	1.234375	1	10.00%
H7	LW=1	10	1.703125	0	0.00%
H7	LW=19.5	10	1.671875	0	0.00%
H7	LW=0.001	10	1.296875	0	0.00%
H8	LW=1	10	1.421875	0	0.00%
H8	LW=19.5	11	1.4375	1	10.00%
H8	LW=0.001	11	1.21875	1	10.00%
H9	LW=1	11	1.4375	1	10.00%
H9	LW=19.5	11	1.421875	1	10.00%
H9	LW=0.001	11	1.234375	1	10.00%
H10	LW=1	11	1.53125	1	10.00%
H10	LW=19.5	11	1.546875	1	10.00%
H10	LW=0.001	12	1.4375	2	20.00%
H11	LW=1	11	0.078125	1	10.00%
H11	LW=19.5	11	0.078125	1	10.00%
H11	LW=0.001	11	0.0625	1	10.00%
H12	LW=1	11	0.53125	1	10.00%
H12	LW=19.5	13	0.4375	3	30.00%
H12	LW=0.001	13	0.390625	3	30.00%
H13	LW=1	13	0.296875	3	30.00%
H13	LW=19.5	13	0.328125	3	30.00%
H13	LW=0.001	13	0.296875	3	30.00%
H14	LW=1	11	0.0625	1	10.00%
H14	LW=19.5	11	0.0625	1	10.00%
H14	LW=0.001	12	0.0625	2	20.00%
H15	LW=1	13	0.421875	3	30.00%
H15	LW=19.5	11	0.5625	1	10.00%
H15	LW=0.001	10	0.40625	0	0.00%

H16	LW=1	13	0.375	3	30.00%
H16	LW=19.5	13	0.328125	3	30.00%
H16	LW=0.001	10	0.3125	0	0.00%
H17	LW=1	11	0.0625	1	10.00%
H17	LW=19.5	11	0.0625	1	10.00%
H17	LW=0.001	12	0.078125	2	20.00%
H18	LW=1	10	0.5	0	0.00%
H18	LW=19.5	13	0.375	3	30.00%
H18	LW=0.001	12	0.546875	2	20.00%
H19	LW=1	10	0.296875	0	0.00%
H19	LW=19.5	13	0.3125	3	30.00%
H19	LW=0.001	11	0.34375	1	10.00%
H20	LW=1	13	0.078125	3	30.00%
H20	LW=19.5	13	0.0625	3	30.00%
H20	LW=0.001	13	0.078125	3	30.00%
H21	LW=1	13	0.359375	3	30.00%
H21	LW=19.5	13	0.390625	3	30.00%
H21	LW=0.001	13	0.359375	3	30.00%
H22	LW=1	13	0.359375	3	30.00%
H22	LW=19.5	13	0.296875	3	30.00%
H22	LW=0.001	13	0.296875	3	30.00%
H23	LW=1	13	0.078125	3	30.00%
H23	LW=19.5	13	0.0625	3	30.00%
H23	LW=0.001	13	0.078125	3	30.00%
H24	LW=1	13	0.390625	3	30.00%
H24	LW=19.5	13	0.4375	3	30.00%
H24	LW=0.001	10	0.375	0	0.00%
H25	LW=1	13	0.296875	3	30.00%
H25	LW=19.5	13	0.296875	3	30.00%
H25	LW=0.001	10	0.296875	0	0.00%
H26	LW=1	13	0.0625	3	30.00%
H26	LW=19.5	13	0.078125	3	30.00%
H26	LW=0.001	13	0.078125	3	30.00%
H27	LW=1	10	0.375	0	0.00%
H27	LW=19.5	13	0.34375	3	30.00%
H27	LW=0.001	10	0.390625	0	0.00%
H28	LW=1	10	0.28125	0	0.00%
H28	LW=19.5	13	0.28125	3	30.00%
H28	LW=0.001	11	0.34375	1	10.00%

Test Instances		T23		
Job Number		4		
Operation Number		11		
Edge Number		407		
Node Number		45		
Total Length		21.5		
Average Length		1.954545		
Optimum Time Slot		10		

	LWs	Makespan	Computation Time (s)	Difference	Error Rate
IP		10	239.8401689	0	0.00%
H1	LW=1	11	2.96875	1	10.00%
H1	LW=21.5	11	2.890625	1	10.00%
H1	LW=0.001	12	1.65625	2	20.00%
H2	LW=1	10	2.140625	0	0.00%
H2	LW=21.5	13	1.78125	3	30.00%
H2	LW=0.001	12	1.515625	2	20.00%
H3	LW=1	13	1.765625	3	30.00%
H3	LW=21.5	13	1.8125	3	30.00%
H3	LW=0.001	10	2.109375	0	0.00%
H4	LW=1	11	2.53125	1	10.00%
H4	LW=21.5	11	2.546875	1	10.00%
H4	LW=0.001	12	1.40625	2	20.00%
H5	LW=1	10	2.453125	0	0.00%
H5	LW=21.5	11	2.53125	1	10.00%
H5	LW=0.001	12	1.640625	2	20.00%
H6	LW=1	10	2.546875	0	0.00%
H6	LW=21.5	11	2.484375	1	10.00%
H6	LW=0.001	12	1.703125	2	20.00%
H7	LW=1	10	2.75	0	0.00%
H7	LW=21.5	10	2.75	0	0.00%
H7	LW=0.001	13	1.65625	3	30.00%
H8	LW=1	10	2.53125	0	0.00%
H8	LW=21.5	11	2.546875	1	10.00%
H8	LW=0.001	10	1.984375	0	0.00%
H9	LW=1	11	2.53125	1	10.00%
H9	LW=21.5	11	2.46875	1	10.00%
H9	LW=0.001	13	1.671875	3	30.00%
H10	LW=1	11	2.625	1	10.00%
H10	LW=21.5	11	2.578125	1	10.00%
H10	LW=0.001	13	1.796875	3	30.00%
H11	LW=1	11	0.140625	1	10.00%
H11	LW=21.5	11	0.125	1	10.00%
H11	LW=0.001	13	0.125	3	30.00%

H12	LW=1	11	0.84375	1	10.00%
H12	LW=21.5	11	0.859375	1	10.00%
H12	LW=0.001	13	0.671875	3	30.00%
H13	LW=1	11	0.671875	1	10.00%
H13	LW=21.5	11	0.71875	1	10.00%
H13	LW=0.001	12	0.578125	2	20.00%
H14	LW=1	11	0.125	1	10.00%
H14	LW=21.5	11	0.171875	1	10.00%
H14	LW=0.001	14	0.125	4	40.00%
H15	LW=1	10	0.78125	0	0.00%
H15	LW=21.5	11	0.953125	1	10.00%
H15	LW=0.001	13	0.65625	3	30.00%
H16	LW=1	10	0.59375	0	0.00%
H16	LW=21.5	13	0.515625	3	30.00%
H16	LW=0.001	12	0.609375	2	20.00%
H17	LW=1	12	0.125	2	20.00%
H17	LW=21.5	11	0.125	1	10.00%
H17	LW=0.001	12	0.125	2	20.00%
H18	LW=1	13	0.609375	3	30.00%
H18	LW=21.5	13	0.703125	3	30.00%
H18	LW=0.001	10	0.828125	0	0.00%
H19	LW=1	13	0.484375	3	30.00%
H19	LW=21.5	13	0.5625	3	30.00%
H19	LW=0.001	10	0.578125	0	0.00%
H20	LW=1	15	0.125	5	50.00%
H20	LW=21.5	15	0.140625	5	50.00%
H20	LW=0.001	15	0.140625	5	50.00%
H21	LW=1	13	0.59375	3	30.00%
H21	LW=21.5	13	0.5625	3	30.00%
H21	LW=0.001	13	0.640625	3	30.00%
H22	LW=1	11	0.671875	1	10.00%
H22	LW=21.5	11	0.65625	1	10.00%
H22	LW=0.001	12	0.5625	2	20.00%
H23	LW=1	15	0.140625	5	50.00%
H23	LW=21.5	15	0.140625	5	50.00%
H23	LW=0.001	15	0.125	5	50.00%
H24	LW=1	10	0.734375	0	0.00%
H24	LW=21.5	13	0.65625	3	30.00%
H24	LW=0.001	12	0.65625	2	20.00%
H25	LW=1	10	0.5625	0	0.00%
H25	LW=21.5	13	0.484375	3	30.00%
H25	LW=0.001	12	0.609375	2	20.00%
H26	LW=1	15	0.140625	5	50.00%

H26	LW=21.5	15	0.140625	5	50.00%
H26	LW=0.001	15	0.140625	5	50.00%
H27	LW=1	13	0.5625	3	30.00%
H27	LW=21.5	13	0.65625	3	30.00%
H27	LW=0.001	10	0.734375	0	0.00%
H28	LW=1	13	0.484375	3	30.00%
H28	LW=21.5	13	0.546875	3	30.00%
H28	LW=0.001	10	0.5625	0	0.00%

Test Instances		T24			
Job Number		5			
Operation Number		12			
Edge Number		603			
Node Number		51			
Total Length		23.5			
Average Length		1.958333			
Optimum Time Slot		10			
	LWs	Makespan	Computation Time (s)	Difference	Error Rate
IP		10	1286.021558	0	0.00%
H1	LW=1	13	4.125	3	30.00%
H1	LW=23.5	13	4.34375	3	30.00%
H1	LW=0.001	14	3.15625	4	40.00%
H2	LW=1	11	4.390625	1	10.00%
H2	LW=23.5	13	4.140625	3	30.00%
H2	LW=0.001	14	2.875	4	40.00%
H3	LW=1	13	3.890625	3	30.00%
H3	LW=23.5	13	3.875	3	30.00%
H3	LW=0.001	11	4.375	1	10.00%
H4	LW=1	11	5.390625	1	10.00%
H4	LW=23.5	11	5.40625	1	10.00%
H4	LW=0.001	14	3.03125	4	40.00%
H5	LW=1	11	4.71875	1	10.00%
H5	LW=23.5	11	5.234375	1	10.00%
H5	LW=0.001	14	3.84375	4	40.00%
H6	LW=1	11	4.796875	1	10.00%
H6	LW=23.5	11	5.390625	1	10.00%
H6	LW=0.001	14	3.78125	4	40.00%
H7	LW=1	12	6.21875	2	20.00%
H7	LW=23.5	12	6.46875	2	20.00%
H7	LW=0.001	14	3.796875	4	40.00%

H8	LW=1	11	5.421875	1	10.00%
H8	LW=23.5	11	5.421875	1	10.00%
H8	LW=0.001	11	4.90625	1	10.00%
H9	LW=1	11	5.1875	1	10.00%
H9	LW=23.5	11	5.28125	1	10.00%
H9	LW=0.001	14	4.734375	4	40.00%
H10	LW=1	11	5.40625	1	10.00%
H10	LW=23.5	11	5.265625	1	10.00%
H10	LW=0.001	13	3.625	3	30.00%
H11	LW=1	11	0.234375	1	10.00%
H11	LW=23.5	11	0.203125	1	10.00%
H11	LW=0.001	15	0.234375	5	50.00%
H12	LW=1	13	1.140625	3	30.00%
H12	LW=23.5	13	1.234375	3	30.00%
H12	LW=0.001	15	1.28125	5	50.00%
H13	LW=1	13	0.84375	3	30.00%
H13	LW=23.5	13	0.859375	3	30.00%
H13	LW=0.001	14	0.921875	4	40.00%
H14	LW=1	13	0.203125	3	30.00%
H14	LW=23.5	11	0.21875	1	10.00%
H14	LW=0.001	16	0.234375	6	60.00%
H15	LW=1	11	1.515625	1	10.00%
H15	LW=23.5	13	0.953125	3	30.00%
H15	LW=0.001	14	1.515625	4	40.00%
H16	LW=1	11	0.953125	1	10.00%
H16	LW=23.5	13	0.890625	3	30.00%
H16	LW=0.001	14	1.109375	4	40.00%
H17	LW=1	13	0.203125	3	30.00%
H17	LW=23.5	11	0.21875	1	10.00%
H17	LW=0.001	12	0.234375	2	20.00%
H18	LW=1	13	1.25	3	30.00%
H18	LW=23.5	13	1.078125	3	30.00%
H18	LW=0.001	11	1.515625	1	10.00%
H19	LW=1	13	0.828125	3	30.00%
H19	LW=23.5	13	0.84375	3	30.00%
H19	LW=0.001	11	0.953125	1	10.00%
H20	LW=1	15	0.25	5	50.00%
H20	LW=23.5	17	0.265625	7	70.00%
H20	LW=0.001	15	0.25	5	50.00%
H21	LW=1	13	1.5625	3	30.00%
H21	LW=23.5	13	1.609375	3	30.00%
H21	LW=0.001	14	1.21875	4	40.00%
H22	LW=1	13	0.890625	3	30.00%

H22	LW=23.5	13	1	3	30.00%
H22	LW=0.001	14	0.9375	4	40.00%
H23	LW=1	15	0.25	5	50.00%
H23	LW=23.5	17	0.265625	7	70.00%
H23	LW=0.001	15	0.25	5	50.00%
H24	LW=1	13	1.59375	3	30.00%
H24	LW=23.5	13	1.4375	3	30.00%
H24	LW=0.001	14	1.484375	4	40.00%
H25	LW=1	11	1	1	10.00%
H25	LW=23.5	13	0.921875	3	30.00%
H25	LW=0.001	14	1.078125	4	40.00%
H26	LW=1	17	0.28125	7	70.00%
H26	LW=23.5	17	0.296875	7	70.00%
H26	LW=0.001	17	0.296875	7	70.00%
H27	LW=1	13	1.3125	3	30.00%
H27	LW=23.5	13	1.125	3	30.00%
H27	LW=0.001	13	1.53125	3	30.00%
H28	LW=1	13	0.84375	3	30.00%
H28	LW=23.5	13	0.859375	3	30.00%
H28	LW=0.001	11	1	1	10.00%

References

Alander, J. T. (2014). An Indexed Bibliography of Genetic Algorithms in Operations Research, Report Series No. 94-1-OR, Retrieved from http://www.uva.fi/~TAU/reports/report94-1/gaORbib.pdf

Alekseev, V. E. (2004). Polynomial algorithm for finding the largest independent sets in graphs without forks. Discrete Applied Mathematics, vol. 135, no. 1–3, pp. 3–16.

Babel, L. (1994). A fast algorithm for the maximum weight clique problem. Computing, 52 (1), 31–38.

Basu, A. T. A. N. U. (2013). Five pillars of prescriptive analytics success. Analytics magazine,

8, 8–12.

Belfares, L., Klibi, W., Lo, N. & Guitouni, A. (2007). Multi-objectives tabu search based algorithm for progressive resource allocation. European Journal of Operational Research, 177, 1779–1799.

Blazewicz, J., Ecker, K.H., Pesch, E., Schmidt, G. & Weglarz, J. (2001). Scheduling Computer and Manufacturing Processes, 2nd edition, Springer-Verlag, Berlin, Germany.

Bloechliger, I. & Zufferey, N. (2013). Multi-coloring and Project-scheduling with Incompatibility and Assignment Costs. Annals of Operations Research, 211, 83-101.

Borassi, M., Crescenzi, P., Habib, M., Kosters, W.A., Marino, A. & Takes, F.W. (2015). Fast Graph Diameter and Radius BFS-Based Computation in (Weakly Connected) Real-World Graphs. Theoretical Computer Science, 586, 59-80. DOI: https://doi.org/10.1016/j.tcs.2015.02.033.

Bron, C. & Kerbosch, J. (1973). Algorithm 457: Finding all cliques of an undirected graph. Communications of the ACM, 16, 575–577.

Case, K. & Wan Harun, W. A. (2000). Feature-based representation for manufacturing planning. International Journal of Production Research, 38 (17), 4285–4300.

Cazals, F. & Karande, C. (2008). A note on the problem of reporting maximal cliques. Theoretical Computer Science, 407 (1-3), 564 - 568.

Chan, F. T. S., Chung, S. H. & Chan, P. L. Y. (2005). An adaptive genetic algorithm with dominated genes for distributed scheduling problems. Expert Systems with Applications, 29 (2), 364–371.

Chan, F. T. S., Chung, S. H. & Chan, P. L. Y. (2006). Application of genetic algorithms with dominant genes in a distributed scheduling problem in flexible manufacturing systems. International Journal of Production Research, 44 (3), 523–543.

Chan, F. T. S., Chung, S. H. & Chan, P. L. Y. (2008). An introduction of dominant genes in genetic algorithm for FMS. International Journal of Production Research, 46 (16), 4369–4390.

Chan, F. T. S., Kumar, V. & Tiwari, M. K. (2009). The relevance of outsourcing and leagile strategies in performance optimization of an integrated process planning and scheduling model. International Journal of Production Research, 47(1), 119–142.

Chaube, A., Benyouef, L. & Tiwari, M. K. (2012). An adapted NSGA-2 based dynamic process plan generation for a reconfigurable manufacturing system. Journal of Intelligent Manufacturing, 23 (4), 1141–1155.

Chen, G. H., Kuo, M. T. & Sheu, J. P. (1988). An optimal time algorithm for finding a maximum weight independent set in a tree. BIT Numerical Mathematics, 28 (2), 353–356.

Chu, C. P. & Gadh, R. (1996). Feature-based approach for set-up minimization of process design from product design. Computer-Aided Design, 26 (5), 321-332.

Czygrinow, A. & Hanckowiak, M. (2006). Distributed algorithms for weighted problems in sparse graphs. Journal of Discrete Algorithms, 4 (4), 588–607.

Dandashi, A. & Al-Mouhamed, M. (2010). Graph Coloring for Class Scheduling. ACS/IEEE International Conference on Computer Systems and Applications - AICCSA 2010, 1, 1-4.

De Werra, D., Demange, M., Monnot, J. & Paschos, V. T. (2005). A hypocoloring model for batch scheduling, Discrete Applied Mathematics, 146 (2005), 3–26.

Diestel, R. (2006), Graph Theory, Graduate texts in mathematics, 173, Springer-Verlag, 3–4, ISBN 9783540261834.

Diestel, R. (2012), "1.9 Some linear algebra", Graph Theory, Graduate Texts in Mathematics. 173, Springer, 23–28.

Du, P. & Zhang, Y. (2016). A New Distributed Approximation Algorithm for the Maximum Weight Independent Set Problem, Mathematical Problems in Engineering, Volume 2016, Article ID 9790629, 10 pages, DOI: http://dx.doi.org/10.1155/2016/9790629.

Duarte, A., Pantrigo, J., Pardo, E. & Mladenovic, N. (2015). Multi-objective variable neighborhood search: An application to combinatorial optimization problems. Journal of Global Optimization, 63 (3), 515–536. DOI: https://doi.org/10.1007/s10898-014-0213-z.

Engel, Y., Etzion, O. & Feldman, Z. (2012). A basic model for proactive event-driven computing. Proceedings of the 6th ACM international conference on distributed event-based systems - DEBS' 12.

Eppstein, D. (2005). All maximal independent sets and dynamic dominance for sparse graphs. Proceedings of the sixteenth annual ACM-SIAM symposium on Discrete algorithms, SODA '05, pages 451-459, Philadelphia, PA, USA. Society for Industrial and Applied Mathematics.

Eppstein, D., Lffler, M. & Strash, D. (2010). Listing all maximal cliques in sparse graphs in near-optimal time. In Cheong, O., Chwa, K.-Y., and Park, K., editors, Algorithms and Computation, volume 6506 of Lecture Notes in Computer Science, pages 403-414. Springer Berlin / Heidelberg.

Epstein, L., Halldórsson, M. M., Levin, A. & Shachnai, H. (2009). Weighted sum coloring in batch scheduling of conflicting jobs. Algorithmica, 55 (2009), 643–665.

Erickson, J. (2018). Solving recurrences. Lecture notes. Retrieved from https://jeffe.cs.illinois.edu/teaching/algorithms/notes/99-recurrences.pdf

Fomin, F. V., Gaspers, S. & Saurabh, S. (2006). Branching and treewidth based exact algorithms. Algorithms and Computation, Editor, Asano, T. 4288 of Lecture Notes in Computer Science, 16–25, Springer, New York, NY, USA.

Furer, M. & Kasiviswanathan, S. (2007). Algorithms for counting 2-SAT solutions and colorings with applications. Proceedings of the 3rd International Conference on Algorithmic Aspects in Information and Management (AAIM ’07), 47–57.

Fukunaga, T., Halldórsson, M. M. & Nagamochi, H. (2007). ‘‘Rent-or-buy’’ scheduling and cost coloring problems. Proceedings of the 27th Int. Conference on Foundations of Software Technology and Theoretical Computer Science, Springer-Verlag, Berlin, Heidelberg, 84-95.

Gainanov D. N., Mladenovic, NENAD & Rasskazova, V. A. (2018) Maximum independent set in planning freight railway transportation Front. Engineering Management, 5 (4), 499–506.

DOI: https://doi.org/10.15302/J-FEM-2018031.

Gamache, M., Hertz, A. & Ouellet, J.O. (2007). A graph coloring model for a feasibility problem in monthly crew scheduling with preferential bidding. Computers & Operations Research, 34 (8), 2384-2395.

Gardi, F. (2009). Mutual exclusion scheduling with interval graphs or related classes, Part I. Discrete Applied Mathematics, 157, 19-35.

Gartner. (2017). Planning guide for data and analytics. Last Accessed: 10 May 2020 https://www.gartner.com/en/documents/3471553/2017-planning-guide-for-data-and-analytics.

Gfeller, B. & Vicari, E. (2007). A randomized distributed algorithm for the maximal independent set problem in growth-bounded graphs. Proceedings of the 26th Annual ACM Symposium on Principles of Distributed Computing (PODC '07), 53–60, Portland, Ore, USA. DOI: https://doi.org/10.1145/1281100.1281111.

Giaro, K., Kubale, M. & Obszarski, P. (2009) A graph coloring approach to scheduling of multiprocessor tasks on dedicated machines with availability constraints, Discrete Applied Mathematics, 157 (2009), 3625–3630.

Grotschel, M., Lovasz, L. & Schrijver, A. (1993). Geometric Algorithms and Combinatorial Optimization, Springer, Berlin, Germany.

Guo, Y. W., Mileham, A. R., Owen, G. W. & Li, W. D. (2006). Operation sequencing optimization using a particle swarm optimization approach, Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture, 220 (12), 1945-1958.

Hansen, P., Mladenovic, N., Todosijevic, R. & Hanafi, S. (2017). Variable neighborhood search:

basics and variants. European Journal on Computational Optimization, 5 (3), 423–454. DOI: https://doi.org/10.1007/s13675-016-0075-x

Halldorsson, M. M. (2000). Approximations of Weighted Independent Set and Hereditary Subset Problems. Journal of Graph Algorithms and Applications, 4 (1), 1-16.

Halldorsson, M. M. (2004). Approximations of independent sets in graphs. Approximation Algorithms for Combinatorial Optimization, 24–45, Springer, Berlin, Germany.

Halldórsson, M. M. & Kortsarz, G. (2004). Multicoloring: Problems and techniques. Editors, Fiala, J., Koubek, V. & Kratochvíl, J., Mathematical Foundations of Computer Science. Lecture Notes in Computer Science, volume 3153, Springer, Berlin, Heidelberg, New York, 21–45.

Huang, F. (2013). On the maximum weighted independent Set Problem with Applications in Wireless Sensor Networks, PhD Thesis, Boston University.

Hansen, P., Mladenovic, N., Todosijevic, R. & Hanafi, S. (2017). Variable neighborhood search: basics and variants. European Journal on Computational Optimization, 5(3), 423–454.

Jia, H. Z., Fuh, J. Y. H., Nee, A. Y. C. & Zheng, Y. F. (2002). Web-based multi-functional scheduling system for a distributed manufacturing environment. Concurrent Engineering, 10 (1), 27–39.

Jia, H. Z., Fuh, J. Y. H., Nee, A. Y. C. & Zhang, Y. F. (2007). Integration of genetic algorithm and Gantt chart for job shop scheduling in distributed manufacturing systems. Computers and Industrial Engineering, 53 (2), 313–320.

Jia, H. Z., Nee, A. Y. C., Fuh, J. Y. H. & Zhang, Y. F. (2003). A modified genetic algorithm for distributed scheduling problems. Journal of Intelligent Manufacturing, 14 (3), 351–362.

Johnson, D. S., Yannakakis, M. & Papadimitriou, C. H. (1988). On generating all maximal independent sets. Information Processing Letters, 27 (3), 119 - 123.

Joo, C., Lin, X., Ryu, J. & Shroff, N. B. (2013). Distributed Greedy Approximation to Maximum Weighted Independent Set for Scheduling with Fading Channels IEEE/ACM Transactions on Networking, 24, 3, 1476–1488. DOI: https://doi.org/10.1109/TNET.2015.2417861.

Kako, A., Ono, T., Hirata, T. & Halldsson, M. (2005). Approximation algorithms for the weighted independent set problem. Proceedings of the 31st International Workshop on Graph-Theoretic Concepts in Computer Science, 341–350.

Karp, R. & Sipser, M. (1981). Maximummatchings in sparse random graphs. Proceedings of the Annual IEEE Symposium on Foundations of Computer Science (FOCS ’81), 364–375, Los Alamitos, California, USA.

Kim, Y. K., Park, K. & Ko, J. (2003). A symbiotic evolutionary algorithm for the integration of process planning and job shop scheduling. Computers and Operations Research, 30, 1151–1171.

Köhler, E. & Mouatadid, L. (2016). A linear time algorithm to compute a maximum weighted independent set on cocomparability graphs. Information Processing Letters, 116 (6), 391-395.

Krarup, J. & De Werra, D. (1982). Chromatic optimization: Limitations, objectives, uses, references. European Journal of Operational Research, 11, 1-19.

Lepenioti, K., Bousdekis, A., Apostolou, D. & Mentzas, G. (2020). Prescriptive analytics: Literature review and research challenges. International Journal of Information Management, 50 (2020), 57-70. DOI: 10.1016/j.ijinfomgt.2019.04.003.

Li, L., Fuh, J., Zhang, Y. & Nee, A. (2005). Application of genetic algorithm to computer-aided process planning in distributed manufacturing environments. Robotics and Computer Integrated Manufacturing, 21 (6), 568–578.

Li, W. & McMahon, C. (2007). A simulated annealing-based optimization approach for integrated process planning and scheduling. International Journal of Computer Integrated Manufacturing, 20 (1), 80–95.

Li, X., Gao, L. & Wen, X. (2013). Application of an efficient modified particle swarm optimization algorithm for process planning. The International Journal of Advanced Manufacturing Technology, 67 (5-8), 1355-1369.

Li, Y. (2018). A smart products lifecycle management (sPLM) framework-modeling for conceptualization, interoperability, and modularity. PhD Thesis, Syracuse University.

Loukakis, E. & Tsouros, C. (1981). A depth first search algorithm to generate the family of maximal independent sets of a graph lexicographically. Computing 27, 349–366. DOI: https://doi.org/10.1007/BF02277184

Lovasz, L. (1994). Stable set and polynomials. Discrete Mathematics, 124, 137–153.

Makino, K. & Uno, T. (2004). New algorithms for enumerating all maximal cliques. In Hagerup, T. and Katajainen, J., editors, Algorithm Theory - SWAT 2004, volume 3111 of Lecture Notes in Computer Science, pages 260-272. Springer Berlin/Heidelberg.

Maropoulos, P. G. & Baker, R. P. (2000). Integration of tool selection with design (part 1. Feature creation and selection of operations and tools). Journal of Material Process Technology, 107, 127–134.

Marx, D. (2004). Graph coloring problems and their applications in scheduling. John Von

Neuman PhD students conference, 48, 11–16.

Matsui, T. (1998). Approximation algorithms for maximum independent set problems and fractional coloring problems on unit disk graphs. Proceedings of the Japan Conference on Discrete and Computational Geometry (JCDCG ’98), 194–200, Tokyo, Japan.

Meuwly, F. X., Ries, B. & Zufferey, N. (2010). Solution methods for a scheduling problem with incompatibility and precedence constraints. Algorithmic Operations Research, 5 (2010), 75–85.

Milosevic, M., Lukic, D., Durdev, M., Vukman, J. & Antic, A. (2016). Genetic algorithms in integrated process planning and scheduling – a state of the art review. Proceedings in Manufacturing Systems, 11 (2), 83–88.

Miner, S.K., Elmohamed, S. & Yau, H. W. (1995). Optimizing Timetabling Solutions using Graph Coloring. NPAC, Syracuse University.

Minty, G. J. (1980). On maximal independent sets of vertices in claw-free graphs. Journal of Combinatorial Theory, Series B, 28 (3), 284–304.

Moon, C. & Seo, Y. (2005). Evolutionary algorithm for advanced process planning and scheduling in a multi-plant. Computers and Industrial Engineering, 48, 311–325.

Moon, J. W. and Moser, L. (1965). On Cliques in Graphs. Israel Journal of Mathematics, 3, 23-28. DOI: http://dx.doi.org/10.1007/BF02760024.

Morad, N. & Zalzala, A. M. S. (1999). Genetic algorithms in integrated process planning and scheduling. Journal of IntelligentManufacturing, 10 (2), 169–179.

Ostergard, P. R. (2002). A fast algorithm for the maximum clique problem. Discrete Applied Mathematics, 120 (1–3), 197–207.

Papadimitriou, C. H. & Steiglitz, K. (1982). Combinatorial Optimization: Algorithms and Complexity, Prentice-Hall, 1982.

Pardalos, P. M. & Xue, J. (1994). The maximum clique problem. Journal of Global Optimization, 4, 301–328.

Paton, K. (1969). An Algorithm for Finding a Fundamental Set of Cycles of a Graph, Communications of the ACM 12, 9, 514-518.

Qiao, L. & Lv, S. (2012). An improved genetic algorithm for integrated process planning and scheduling. International Journal of Advanced Manufacturing Technology, 58 (5), 727–740.

Sakai, S., Togasaki, M. & Yamazaki, K. (2003). A note on greedy algorithms for the maximum weighted independent set problem, Discrete Applied Mathematics, 126, 313-322.

Salehi, M. & Bahreininejad, A. (2011). Optimization process planning using hybrid genetic algorithm and intelligent search for job shop machining. Journal of Intelligent Manufacturing, 22 (4), 643–652. DOI: 10.1007/s10845-010-0382-7.

Sharma, G., Shroff, N. B. & Mazumdar, R. R. (2006). On the Complexity of Scheduling in Wireless Networks. ACM MOBICOM.

Spall, J. C. (2003). Introduction to Stochastic Search and Optimization. Wiley. ISBN 978-0-471-33052-3.

Su, Y., Chu, X., Chen, D. & Sun, X. (2018). A genetic algorithm for operation sequencing in CAPP using edge selection based encoding strategy. Journal of Intelligent Manufacturing, 29:313–332.

Sun, K., Li, Y. & Roy, U. (2017). A PLM-based data analytics approach for improving product development lead time in an engineer-to-order manufacturing firm. Mathematical Modelling of Engineering Problems, 4 (2), 69-74. DOI: 10.18280/mmep.040201.

Takes, F. W. & Kosters, W.A. (2011). Determining the Diameter of Small World, Networks. Proceedings of the 20th ACM International Conference on Information and Knowledge Management (CIKM 2011), 1191-1196. DOI: https://doi.org/10.1145/2063576.2063748.

Takes, F. W. & Kosters, W.A. (2013). Computing the Eccentricity Distribution of Large Graphs. Algorithms, 6(1), 100-118. DOI: https://doi.org/10.3390/a6010100.

Tassiulas, L. & Ephremides, A. (1992). Stability properties of constrained queueing systems and scheduling policies for maximal throughput in multihop radio networks. IEEE Transactions on Automatic Control, 37 (12), 1936-1948.

Thevenin, S., Zufferey, N. & Potvin, J. (2018). Graph multi-coloring for a job scheduling application. Discrete Applied Mathematics, 234, 218–235

Todosijevic, R. & Mladenovic, N. (2016). Nested general variable neighborhood search for the periodic maintenance problem. European Journal of Operational Research, 252 (2), 385–396.

Tomita, E., Tanaka, A. & Takahashi, H. (2006). The worst-case time complexity for generating all maximal cliques and computational experiments. Theoretical Computer Science, 363 (1), 28-42.

Tucker, A. (2012). Applied Combinatorics. John Wiley & Sons, Inc, 6th edition.

Valiente, G. (2003). A New Simple Algorithm for the Maximum-Weight Independent Set Problem on Circle Graphs. In: Ibaraki T., Katoh N., Ono H. (eds) Algorithms and Computation. ISAAC 2003. Lecture Notes in Computer Science, vol 2906. Springer, Berlin, Heidelberg.

Weisstein, E. W. (2020). Graph Diameter. MathWorld--A Wolfram Web Resource, Retrieved from http://mathworld.wolfram.com/GraphDiameter.html.

Zais, M. & Laguna, M. (2016). A graph coloring approach to the deployment scheduling and unit assignment problem. Journal of Scheduling, 19, 73–90. DOI: 10.1007/s10951-015-0434-0

Zhang, F., Zhang, Y. & Nee, A. (1997). Using genetic algorithms in process planning for job shop machining. IEEE Transactions on Evolutionary Computation, 1 (4), 278–289.

Zhang, W. & Gen, M. (2010). Process planning and scheduling in distributed manufacturing system using multiobjective genetic algorithm. IEEJ Transactions on Electrical and Electronic Engineering, 5 (1), 62–72.

Zhang, W., Gen, M. & Jo, J. (2014). Hybrid sampling strategy-based multi-objective evolutionary algorithm for process planning and scheduling problem. Journal of Intelligent Manufacturing 25, 881–897.

Zhang, X. & Yan, H. (2005). Integrated optimization of production planning and scheduling for a kind of job-shop. International Journal of Advanced Manufacturing Technology, 26 (7), 876–886.

Zhang, Y. F., Saravanan, A. N. & Fuh, J. Y. H. (2003). Integration of process planning and scheduling by exploring the flexibility of process planning. International Journal of Production Research, 41 (3), 611-628. DOI: 10.1080/0020754021000037874.

VITA

KAI SUN

211 Lafayette Road, Apt 607 Email: kasun@syr.edu
Syracuse, NY 13205 Phone: +1 315-751-5150

EDUCATION

Doctor of Philosophy in Mechanical Engineering 08/2015 – 08/2020
College of Engineering & Computer Science, Syracuse University

Master of Science in Mechanical & Aerospace Engineering 09/2013 - 05/2015
College of Engineering & Computer Science, Syracuse University

Bachelor of Engineering in Vehicle Engineering 09/2009 - 07/2013
School of Mechanical and Automobile Engineering, Hefei University of Technology

PROFESSIONAL EXPERIENCE

College of Engineering & Computer Science, Syracuse University
Teaching Assistant Fellowship 09/2018 - 05/2020
Teaching and tutoring in courses/labs:

- MAE 284: Introduction to CAD
- MAE 333: Data Analysis for Engineers
- MEE 431: Manufacturing Processes
- MAE 548: Engineering Economics and Technology Valuation

Research Assistant Fellowship 05/2017 - 12/2017
Development and customization of product life-cycle management system (Aras Innovator) for Filtertech, Inc. (Sponsored by the CASE center at Syracuse University)

Research Assistant 10/2016 - 3/2017
Projects:

- Development of Integrated System for Design Operation for UAVs
- Development of Educational and Training Materials for Unmanned Aerial Systems (UAS)

Graduate Assistant 09/2015 - 05/2017
Teaching and tutoring in courses/labs:

- MAE 184: Engineering Graphics and CAD
- MEE 571: Computer-Aided Design
- MFE 639: CAD/CAM Systems
- MFE 692: Design for Manufacturing

UsPLM, Inc.
Mechanical & Research Engineer Internship 05/2018 - 08/2018
Development of digital twin for the drone fleet management, web-based flight simulation

and analysis in 3D virtual reality.

Filtertech, Inc.

Research Assistant Internship 06/2016 - 08/2016

Development and customization of product life-cycle management system (Aras Innovator) with graphical analysis tools.

ACADEMIC PUBILICATIONS

1. Sun, K., & Roy, U. (to be submitted). An algorithm structure for solving maximum weighted independent set problem. Discrete Applied Mathematics.
2. Sun, K., & Roy, U. (to be submitted). Solving the process planning and scheduling problem via maximum weighted independent set. Discrete Applied Mathematics.
3. Sun, K., Li, Y. & Roy, U. (2017). A PLM-based data analytics approach for improving product development lead time in an engineer-to-order manufacturing firm, *Mathematical Modelling of Engineering Problems*, Vol. 4, No. 2, June 2017, pp. 69-74. DOI: 10.18280/mmep.040201

PROJECTS

(a) “Development of the Data Analytics Services for Smart Product Design and Manufacturing Activities Based on a Smart Product Lifecycle Management Platform,” funded by the National Institute of Standards & Technology (NIST);

(b) “Developing a unified lifecycle management platform for smart manufacturing systems, an essential tool for cyber-manufacturing for Filtertech Co.,” funded by the CASE (Center for Advanced Systems and Engineering) Center at Syracuse University;

(c) “Development of Integrated System for Design Operation for UAVs,” funded by the New York State Department of Economic Development (through Gryphon Sensors and SU).

(d) “Development of Educational and Training Materials for Unmanned Aerial Systems (UAS),” funded by the New York State Department of Economic Development (through Gryphon Sensors and SU);

(e) “Optimizing the Task Assignments of Designers for Concurrent Projects in Filtertech Inc,” funded by the SyracuseCoE (New York State's Center of Excellence in Environmental and Energy Systems);

(f) “Digital Twin Development for the UsPLM Drone Fleet Management Solution,” (worked as an intern with the UsPLM, Inc. through CASE Center).

PROFESSIONAL SKILLS

- CAD/CAM: SolidWorks, CATIA V5, PTC CREO, SolidCAM, Fusion 360
- Modeling and Simulation: Star-CCM+, Pointwise, Matlab, Arena, AnyLogic
- Data Analytics: Python, R, KNIME, RapidMiner
- Programming: Python, C#, JavaScript, HTML, C/C++
- Languages: English, Mandarin