



UNIVERSITAS INDONESIA



UNIVERSITÉ D'ARTOIS

**MINIMASI PENALTI ATAS KECEPATAN DAN
KETERLAMBATAN DENGAN MENGGUNAKAN METODE
METAHEURISTIK ELEKTRO MAGNETISM**

TESIS

**Diajukan sebagai salah satu syarat untuk mendapatkan gelar Magister
Teknik**

RIVAI CHANDRA JUNIANTO

0906644386

FAKULTAS TEKNIK

PROGRAM STUDI TEKNIK SIPIL

DEPOK

JULI 2011



UNIVERSITAS INDONESIA



UNIVERSITÉ D'ARTOIS

**MINIMASI PENALTI ATAS KECEPATAN DAN
KETERLAMBATAN DENGAN MENGGUNAKAN METODE
METAHEURISTIK ELEKTRO MAGNETISM**

TESIS

RIVAI CHANDRA JUNIANTO

0906644386

**FAKULTAS TEKNIK
PROGRAM STUDI TEKNIK SIPIL**

DEPOK

JULI 2011


HALAMAN PERNYATAAN ORISINALITAS

Tesis ini adalah hasil karya saya sendiri,
Dan semua sumber baik yang dikutip maupun yang dirujuk
Telah saya nyatakan dengan benar:

Nama : Rivai Chandra Junianto

NPM : 0906644386

Tanda tangan

: 

Tanggal : Juli 2011

Internship Report

In order to obtain, the Master 2 E-Logistics

Faculty of Applied Sciences

University of Artois

Have been presented and defended by:

RIVAI CHANDRA JUNIANTO

On July 5, 2011

Title:

**MINIMIZING EARLINESS AND TARDINESS PENALTIES USING
ELECTROMAGNETISM-LIKE MECHANISM METAHEURISTICS**

Mentors:

Prof. Gilles Goncalves :

Dr. Hamid Allaoui :

Jury:

Prof. Daniel Jolly :

Dr. Hamid Allaoui :

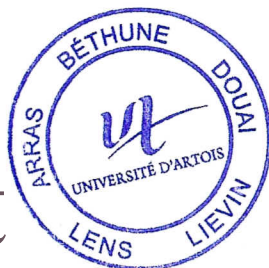
Dr. Tinte Hsu :

Assigned in Bethune – France

On July 8, 2011

Director of Department Industrial Engineering and Logistic

Alain Malesys



~g"t

UCAPAN TERIMA KASIH

Puji syukur saya panjatkan kepada Tuhan Yesus Kristus, atas segala Hikmat dan karunia yang diberikan-Nya sehingga saya dapat menyelesaikan laporan ini. Saya juga tidak lupa mengucapkan terima kasih yang sebesar-besarnya kepada :

1. Badan Diklat Kemenhub yang memberikan saya beasiswa sehingga mendapatkan kesempatan untuk dapat melanjutkan kuliah ke jenjang S2
2. Prof Suyono Dikun atas bimbingan nya selama menjalani kuliah semester 1 dan 2 di Universitas Indonesia
3. Prof Irwan Katili atas kesempatan yang diberikan untuk dapat mengikuti program Double Degree Indonesia-Perancis
4. Prof. Gilles Goncalves dan DR. Hamid Allaoui atas bimbingan yang telah diberikan kepada saya untuk mempelajari Electromagnetism-Like Mechanism dan Program C++ yang merupakan komponen utama dalam penyelesaian tesis ini
5. Keluargaku yang tercinta atas segala kebahagiaan dan dukungan yang diberikan kepada saya selama belajar di Perancis
6. Teman-teman DDIP-2010 khusus nya Grup Bethune atas segala kerjasama dan dukungannya

Depok, 20 Juli 2011

Penulis

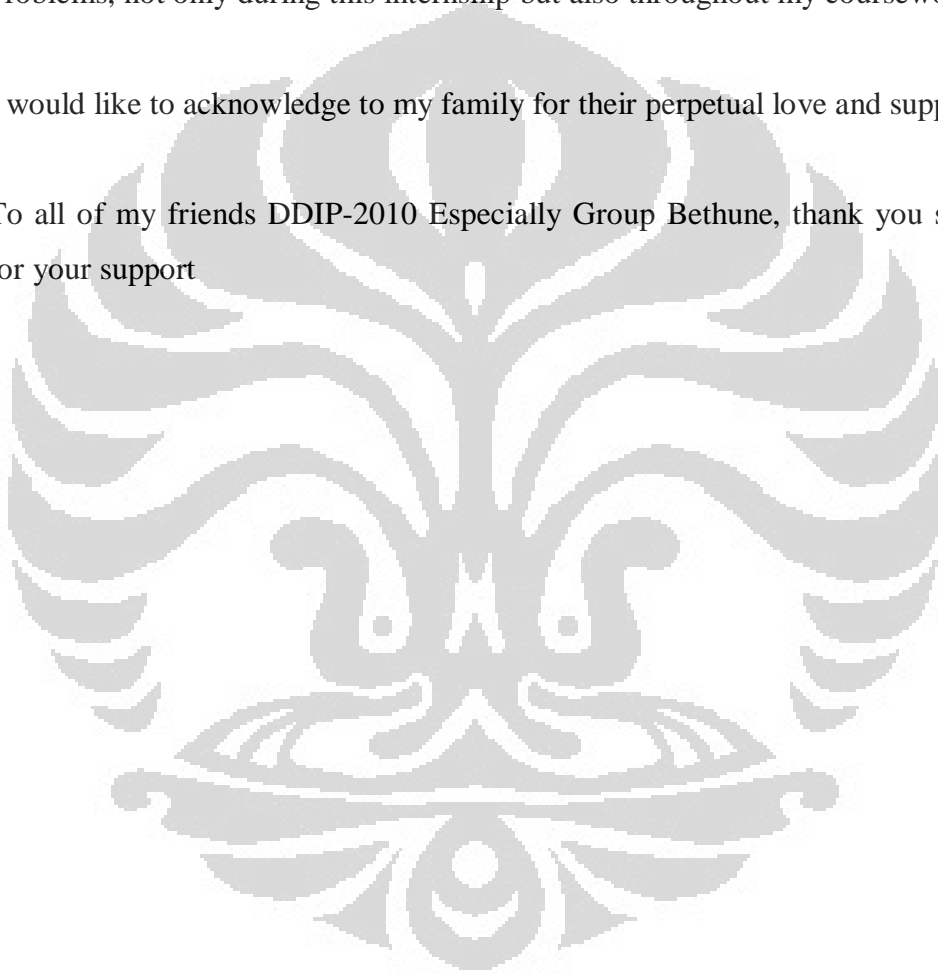
ACKNOWLEDGEMENTS

First and foremost I would like say thanks to **Gilles Goncalves** for his kindness to allow me studying Program of C++ and Electromagnetism-Like Mechanism in LGI2A. This is a key component of this project.

I would like to acknowledge **Hamid Allaoui** for his knowledge about Scheduling Problems, not only during this internship but also throughout my coursework

I would like to acknowledge to my family for their perpetual love and support

To all of my friends DDIP-2010 Especially Group Bethune, thank you so much for your support



**HALAMAN PERNYATAAN PERSETUJUAN PUBLIKASI
TUGAS AKHIR UNTUK KEPENTINGAN AKADEMIS**

Sebagai civitas akademik Universitas Indonesia, Saya yang bertanda tangan dibawah ini :

Nama : Rivai Chandra Junianto
NPM : 0906644386
Program studi : Manajemen Infrastruktur
Departemen : Teknik Sipil
Jenis karya : Tesis

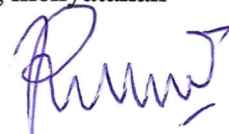
Demi pengembangan ilmu pengetahuan, menyetujui untuk memberikan kepada Universitas Indonesia hak bebas royalti non eksklusif (non-exclusive royalty-free right) atas karya saya yang berjudul : *Minimasi penalti atas kecepatan dan keterlambatan dengan metode heuristik elektro magnetism* beserta perangkat yang ada (jika diperlukan). Dengan hak bebas royalti noneklusif ini Univesitas Indonesia berhak menyimpan, mengalihmedia/formatkan. Mengolah dalam bentuk pangkalan data (database), merawat dan mempublikasikan tugas akhir saya tanpa meminta izin dari saya selama tetap mencantumkan nama saya sebagai penulis/pencipta dan sebagai pemilik hak cipta.

Demikian pernyataan ini saya buat dengan sebenarnya

Dibuat di : Depok

Pada tanggal : 8 Juli 2011

Yang menyatakan



Rivai Chandra Junianto

ABSTRAK

Dalam beberapa dekade terakhir, beberapa algoritma yang efektif untuk memecahkan masalah penjadualan dengan sumber daya yang terbatas telah diusulkan. Hal ini berdasarkan pentingnya masalah optimasi kombinatorial untuk ilmiah dan dunia industri. Bagaimanapun tantangan yang dialami dirangkum dalam satu kuatnya NP-hard, membatasi efektivitas optimasi yang tepat untuk kasus yang relatif kecil. Elektromagnetism adalah populasi berbasis metaheuristik yang telah diusulkan untuk memecahkan masalah efektivitas yang terus-menerus. Untuk memecahkan masalah ini kami menggunakan metaheuristik yang baru yang mengaplikasikan metodologi elektromagnetism untuk memecahkan masalah meminimalkan penalti atas kecepatan dan keterlambatan yang terjadi. Sepengetahuan kami, hanya ada sedikit penelitian untuk memecahkan masalah optimasi kombinatorial dengan elektromagnetism. Penelitian ini mencoba menggunakan konsep acak menggabungkan dengan elektromagnetism algoritma untuk mendapatkan skedul yang terbaik atau optimal untuk masalah meminimalkan penalti atas kecepatan dan keterlambatan yang terjadi. Pendekatan ini mencoba untuk mencapai efek konvergensi dan keragaman ketika proses iteratif diterapkan untuk memecahkan masalah.

Kata kunci : skeduling, elektromagnetism, program C++

ABSTRACT

In the last few decades, several effective algorithms for solving the resource-constrained project scheduling problem have been proposed. This is due to the importance of combinatorial optimization problems for the scientific as well as the industrial world. However, the challenging nature of this problem, summarised in its strongly NP-hard status, restricts the effectiveness of exact optimisation to relatively small instances. Electromagnetism-like algorithm (EM) is a population-based meta-heuristic which has been proposed to solve continuous problems effectively. To solving this problem we using a new meta-heuristic that applies the EM methodology to minimizing aeliness and tardiness penalties problem. To the best of our knowledge, there are only few researches in solving the combinatorial optimization problem (COP) by EM. This research attempts to employ the random-key concept combining with Electromagnetism algorithm to obtain the best/optimal schedule for minimizing aeliness and tardiness penalties problems. This approach attempts to achieve the convergence and diversity effects when it is iteratively applied to solve the problem

Keywords: Scheduling, Elektromagnetism, C++'s program

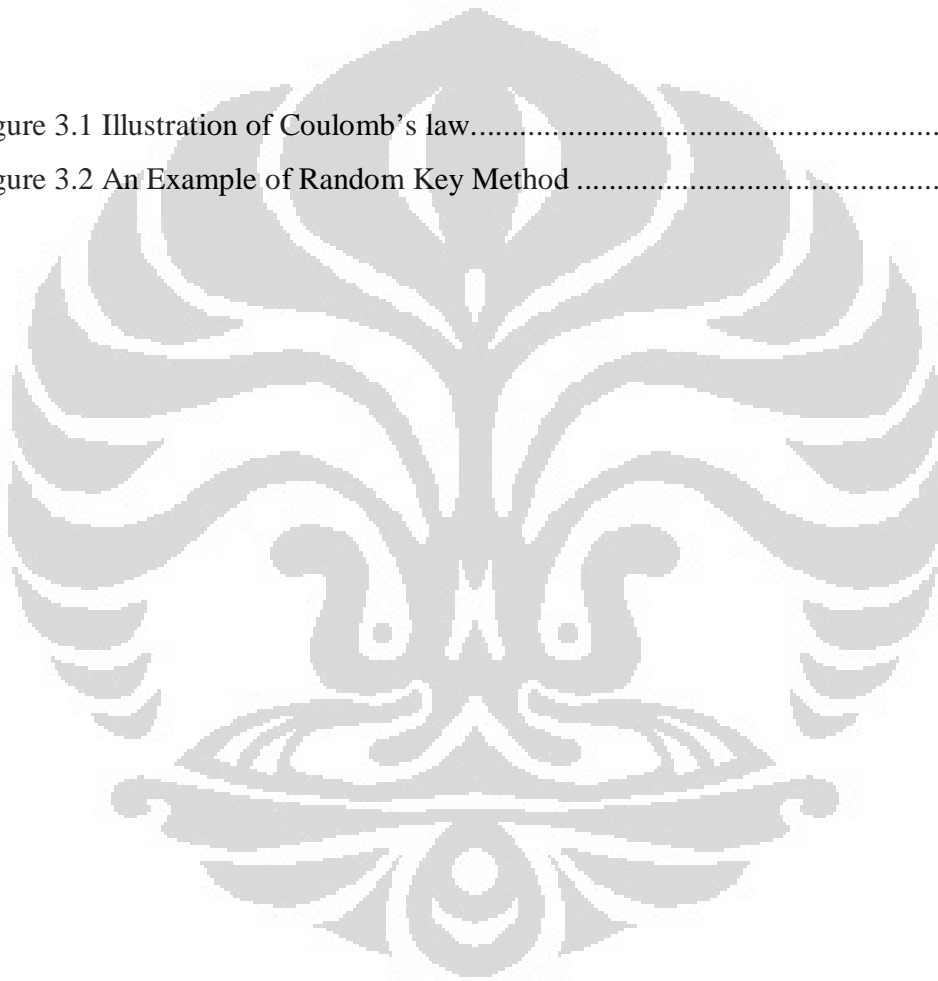
TABLE OF CONTENTS

HALAMAN DEPAN	
HALAMAN PERYATAAN ORISINALITAS	
LEMBAR PENGESAHAN	
UCAPAN TERIMA KASIH	
HALAMAN PERNYATAAN PERSETUJUAN PUBLIKASI	
ABSTRAK	
TABLE OF CONTENTS	i
LIST OF FIGURES	ii
CHAPTER 1 INTRODUCTION	1
1.1 Background.....	1
1.2 The Objective.....	2
1.3The Problem statement.....	3
1.4The Writing Organization.....	3
CHAPTER 2 LITERATURE REVIEW	4
2.1 Metaheuristics for optimization	4
2.2 An Electromagnetism-like Mechanism (EM) Metaheuristics	7
2.3 Earliness and Tardiness penalties problem.....	8
2.4 C++ Programming Language	10
CHAPTER 3 RESOLUTION METHODE	12
3.1 Electromagnetism-like Mechanism Algorithm	12
3.1.1 General Scheme.....	12
3.1.2 Initialization.....	13
3.1.3 Local Search.....	13
3.1.4 Calculation Force	15
3.1.5 Movement	17
3.2 Electromagnetism-like Mechanism Algorithm for Earliness and Tardiness Penalties Problem	18
3.2.1 Main Algorithm.....	19

3.2.2 Initialization.....	19
3.2.3 Local Search.....	20
3.2.4 Particle charges, electrostatic force and move	21
CHAPTER 4 IMPLEMENTATION.....	23
4.1. Initialization Procedure	23
4.2 Local Search Procedure	26
4.3 Calculation Procedure	27
4.4 Analysis.....	28
4.5 Main Program.....	28
4.6. Program Set-up.....	28
CHAPTER 5 EXPERIMENTAL STUDY.....	30
5.1. Entry Data	30
5.2 Result and Comparison	32
CHAPTER 6 CONCLUSION AND PERSPECTIVES.....	33
6.1. Conclusion.....	33
6.2 Perspectives	33
REFERENCES.....	34
APPENDIX.....	35

LIST OF FIGURES

Figure 3.1 Illustration of Coulomb's law.....	15
Figure 3.2 An Example of Random Key Method	19



LIST OF TABLES

Table 3.1 Main Algorithm of EM	12
Table 3.2 Algorithm Initialize	13
Table 3.3 Algorithm Local Search	14
Table 3.4 Algorithm Calculation Force	16
Table 3.5 Algorithm Move	17
Table 3.6 EM Main Algorithm For Minimizing Earliness and Tardiness	20
Table 3.7 EM Initialize Algorithm For SMSP	20
Table 3.8 EM Local Search Algorithm For SMSP	20
Table 3.9 EM Movement Algorithm For SMSP	21
Table 3.10 EM Check Boundary Algorithm For SMSP	22
Table 4.1 Establishing a Random Key with C++ Pseudo Code	24
Table 4.2 Sorting Ascending Pseudo Code	24
Table 4.3 Calculate Earliness, Tardiness and Function Objective Pseudo Code ..	25
Table 4.4 Searching Minimum Function Objective Value Pseudo Code	25
Table 4.5 Defining y_j in Local Search Procedure Pseudo Code	26
Table 4.6 Calculating Charge and Force Pseudo Code	27
Table 4.7 Movement x^i Pseudo Code	28
Table 4.8 Main Program Pseudo Code	28
Table 4.9 Example Program set-up for 5 jobs	29
Table 5.1 Data Entry for 5 jobs	30
Table 5.2 Data Entry for 10 jobs	31
Table 5.3 Data Entry for 15 jobs	31
Table 5.4 Data Entry for 20 jobs	31
Table 5.5 The comparison Result between EM, EDD, SPT and LPT	32

CHAPTER I

INTRODUCTION

1.1 Background

Scheduling is the allocation of scarce resources to activities over time. The resources take form of the machine and activities take form of jobs in a generic manufacturing environment. In our day to day lives, we make decisions that we believe can maximize or minimize our objectives, such as taking a shortcut to minimize the time or distance required to reach a particular destination, or finding a lowest priced items in the supermarket. Most of these decisions are based on our years of experience and knowledge about the system without resorting to any systematic mathematical formulation. However, as the system becomes more complicated, further it is needed to formulate it into specific mathematical model, and with the advent of computer it is possible to exploit optimization theories to their maximum extent. Combinatorial Optimization is a branch of optimization that arises every where and certainly in applied mathematics and computer science, related to operations research, algorithm theory that sit at the intersection of several fields, including artificial intelligence, mathematics and software engineering. Combination optimization algorithm solve problems instances that are believed to be hard in general, by exploring the usually large solution space of these instances. One of problem in the optimization global are minimizing earliness and tardiness penalties problem.

Minimizing earliness and tardiness penalties problem are one of the classical combinatorial optimisation problems which exist in many diverse areas such as flexible manufacturing systems, production planning, air lane industry, etc. Scheduling is the allocation of scarce resources to activities over time. The resources take form of the machines and the activities take form of the jobs in a generic manufacturing environment. Tardiness Penalties is defined as the cost which charged because jobs completed over it's due date and earliness penalties is defined as cost which charged because jobs complete before its due date. Due to its practical importance, there is enormous amount of research in many kinds of minimizing earliness and tardiness problems. Algorithms developed for the single machine model provide a basis for design of exact algorithms and heuristics for

more complicated machine environments. minimizing earliness and tardiness problem with the objective to minimise the total penalties of earliness/tardiness is shown to be NP-hard in the literature.

The results derived from the literature are very significant since they not only provide the insight into the single machine problem but also for more complicated environment (Pinedo,2002).In this study we apply the random key approach to represent a schedule and incorporate the Electromagnetism methodology to minimizing earliness and tardiness penalties problem in our algorithm, the electromagnetism procedures are modified to obtain the better quality of solution effectively. For example, the local search operator perturbs the best solution and generates a new solution. As long as the new one with a better solution than the worst one, we will replace the worst one with the new one. In this paper we also using a new method in calculating the particle charge and exertion force proposed by Debels, Reyck, Leus, and Van Houcke (2006). Finally we will using a programming computer namely program C++ to make logic of all mathematical algorithm in electromagnetism like mechanism.

1.2 The Problems Statement

EM methodology are applied to minimizing earliness and tardiness penalties problem and the objective is to minimize the total penalties of earliness and tardiness. A detailed formulation of the problem is described as follows: a set of n independent jobs (J_1, J_2, \dots, J_n) with positive processing time p_j has to be scheduled without preemptions on a single machine that can handle at most one job at a time and should and has a due date d_j .

The machine is assumed to be continuously available from time zero onwards and unforced machine idle time is not allowed. For any given schedule, the earliness and tardiness of J_j can be, respectively, defined as $E_j = \max(0, d_j - C_j)$ and $T_j = \max(0, C_j - d_j)$, where C_j is the completion time of J_j . The objective is then to find a schedule that minimizes the sum of the earliness and tardiness penalties of all jobs $\sum_{j=1}^n (\alpha_j E_j + \beta_j T_j)$, where α_j and β_j are the earliness and tardiness penalties of job J_j .

The inclusion of both earliness and tardiness costs in the objective function is compatible with the philosophy of just-in-time production, which emphasizes producing

goods only when they are needed. The early cost may represent the cost of completing a product early, the deterioration cost for a perishable goods or a holding (stock) cost for finished goods. The tardy cost can represent rush shipping costs, lost sales and loss of goodwill. It is assumed that no unforced machine idle time is allowed, so the machine is only idle if no job is currently available for processing.

So, the problems of this paper are :

- how to make sequence of jobs (schedule) based on value of activities
- how to determine the total of earliness and tardiness of each jobs
- how to change the job sequence which less optimal become more optimal by using EM algorithm

1.3 The Objectives

The objectives of this paper are to implementing Electromagnetism-like Mechanism (EM) algorithm for minimizing earliness and tardiness penalties problem solving as find a schedule that minimizes the sum of earliness and tardines penalties of all jobs.

1.4 The Writing Organization

The rest of the paper is organized as follows: Chapter I is the review of introduction consist of the background and the probleme statement of earliness and tardiness penalties problem ; Chapter II is the literature review of Metaheuristics for global optimization, Electromagnetism-like Mecanism Meta Heuristics for global optimization and solving earliness and tardiness penalties problem , and review of earliness and tardiness penalties problem ; Chapter III describing Resolution Methode. The Implementation is decribed in Chapter IV; so, The Experimental study is presented in sectin V which compared solution between EM, EDD, SPT and LPT; Chapter VI draws conclusion and Prespectives.

CHAPTER II

LITERATURE REVIEW

2.1 Metaheuristics for Optimization

The term of Metaheuristics is firstly introduced by Fred Glover, derives from the composition of two greek words. Heuristics derives from the verb heuriskein which means “to find” , while the suffix meta means “beyond, in an upper level”. Metaheuristics are designed to tackle complex optimization problems where other optimization methods have failed to be either effective or efficient. These methods have come to be recognized as one of the most practical approaches for solving many complex problems, and this is particularly true for the many real-world problems that are combinatorial in nature. The practical advantage of metaheuristics lies in both their effectiveness and general applicability. In this review, we just discuss heuristics algorithm to understanding global optimization methods, namely Metaheuristics algorithms which are inspired by careful observations of natural phenomena and some of them are developed merely by implementing practical ideas. These algorithm are the Genetic Algorithm (GA), the Ant Colony System (ACS), simulated annealing (SA), the Tabu Search (TS), and the Electromagnetism-like Mechanism (EM).

Genetic Algorithm (GA) was developed by Holland and his student at the University of Michigan in the ‘60s and ‘70s, GA simulates those processes in natural populations which are essential to evolution. Basically, a GA operates on a finite population of chromosomes or bit string. The search mechanism consist of three different phases: evaluation of the fitness of each chromosomes , selection of the parent chromosomes and application of the mutation and recombination operators to the parent chromosomes. The new chromosomes are resulting from these operation from the next generation, and the process is repeated until the system ceases to improve (Potvin, 1996). The pure Genetic Algorithm developed by Holland and his student was not designed to solve combinatorial optimization problem. In the past two decade, many researchers modified the original Genetic Algorithm to handle combinatorial optimization problems.

Ant Colony Optimization (ACO) is a metaheuristic approach proposed in Dorigo 1992, 1996, 1999. The inspiring source of ACO is the foraging behavior of real ants. This

behavior enables ants to find shortest paths between food sources and their nest. Even if a single ant has only restricted abilities, the behaviour of a whole ant colony is highly structured as the consequence of coordinated interactions. The way of ant's communication is a chemical compound, known as pheromone. A moving ant lays various quantities of these compounds on the ground, thus it marks the path its moves with a pheromone trace. Usually a single ant begins to move randomly, but by detecting a pheromone trail, the ant will follow it with higher probability intensifying it with its own pheromone. Hence the probability that ants will follow a path correlates with the number of ants that did so before. This is a form of *autocatalytic* behaviour — or *allelomimesis*. Pheromones also vanish with time if they are not refreshed. If all of the food is taken away from a particular place, ants will stop putting pheromones onto the respective track since they cannot find any food at this location anymore. The process is thus characterized by a positive feedback loop. It starts with a given path from the ant hill to food source. If this path is cut off by an obstacle, the ants have to pass the obstacle along the right or left path. Each ant makes a choice on the basis of some heuristic evaluations and the intensity of the pheromone trails left by previous ants. The path with a good heuristic evaluation and a high level of pheromone is more likely to be selected, so it gives the following ant a stronger stimulus and thus a higher probability to turn right. The first ant forced to decide which path must to be taken has the same possibility to turn right or left (since there is no pheromone trail on the alternative paths). If the right path is shorter than left one, the ant that took it will be faster than the ant following the left (long) path. Due to the shorter distance they can move more often. The next ants will find a stronger trail on the right path and it will become preferred (in probability) to the left path. The probability with which an ant decides to move along the path to follow is more and more prejudiced towards the shorter one because the intensity of pheromones increases faster on the shorter path. The final result is that all ants will quick-witted in choosing of the shorter right path. However, it is significant that the decision is never deterministic, thus there remains always a possibility to explore alternative routes

Simulated Annealing (SA) is commonly said to be the oldest among the metaheuristics and surely one of the first algorithms that has an explicit strategy to escape from local minima. The origins of the algorithm are in statistical mechanics (Metropolis

algorithm) and it was first presented as a search algorithm for CO problem in Kirkpatrick et al. (1983) (given some assumptions on the cooling schedule of the temperature, etc.) that could be shown to converge to an optimal solution. The fundamental idea is to allow moves resulting in solutions of worse quality than the current solution (uphill moves) in order to escape from local minimal. The probability of doing such a move is decreased during the search. Simulated Annealing comes from the annealing process of solids. A solid is heated until it melts, and then the temperature of the solid is slowly decreased (according to annealing schedule) until the solid reaches the lowest energy state or the ground state. If the initial temperature is decreased rapidly, the solid at the ground state will have many defects or imperfections. An easy implementation of the algorithm makes it very easy to adapt a local search method (e.g. best improvement local search) to a simulated annealing algorithm, usually rendering the local search with much better results. But although it is proven to converge to the optimum, it converges in infinite time. Not only for this reason, but also since we have to cool down slowly, the algorithm is usually not faster than its contemporaries.

Tabu Search (TS) was proposed by Glover in 1986. A description of the method and its concepts can be found in Glover and Laguna (1997). The basic principle of TS is to pursue a best improvement Local Search whenever it encounters a local minimum by allowing non-improving moves, cycling back to previously visited solutions is prevented by the use of memories called tabu lists that record the recent history of the search, a key idea that can be linked to Artificial Intelligence concepts. It is also important to remark that Glover did not see TS as a proper heuristic, but rather as a metaheuristic, i.e., a general strategy for guiding and controlling “inner” heuristics specifically tailored to the problems at hand (Gendreau, 2002). Tabu Search (TS) is among the most cited and used metaheuristics for combinatorial problems. Many computational experiments have shown that TS has now become an established optimization technique which can compete with almost all known techniques and which – by its flexibility – can beat many classical procedures. The word tabu (or taboo) comes from Tongan, a language of Polynesia, where it was used by the aborigines of Tonga Island to indicate things that can not be touched because they are sacred. According to Webster’s Dictionary, the word now also means “a prohibition imposed by social custom as a protective measure: or of something

“banned as constituting a risk”. These current more pragmatic senses of the word accord well with the theme of tabu search. The risk to be avoided in this case is that of following a counter-productive course, including one which may lead to entrapment without hope of escape. On the other hand, as in the broader social context where “protective prohibitions” are capable of being superseded when the occasion demands, the “tabus” of tabu search are to be overruled when evidence of a preferred alternative becomes compelling. And Electromagnetism-like Mechanism (EM) Heuristics will explain in sub section follow.

2.2 An Electromagnetism-like Mechanism (EM) Heuristic

The Electromagnetism-like mechanism (EM), the latest algorithm for optimization global was developed by Birbil and Fang in 2003. They make a mechanism that sample point (solution) are with the charge that relate to the objective function value and encourages them to converge to the highly attractive valleys, and contrarily, discourages the sample point (solution) to move further away from steeper hills, it means EM start from randomly selected point from the feasible region to obtain optimization problem. EM employs an attraction-repulsion mechanism to move point (particle) towards the optimal solution. Each particle (point) is treated as a solution and has a charge, this charge related to the objective function value associated with the solution. As in evolutionary search algorithms, a population, or set of solutions is created in which each selection point will exert attraction or repulsion on other point, the magnitude of which is proportional to the product of the charges and inversely proportional to the distance between the point (Coulomb's Law). In this study, it is shown that EM is able to converge to the optimal solution in less number of function evaluations without any first or second order derivative information. In 2004 Birbil et al make a thorical study of this EM analysis modification for convergence to the optimum solving problem. However, these above two studies only deal with continuous optimization problems. EM also can be used to solve fuzzy relation equations (Birbil & Feyzioglu, 2003), train artificial neural network for textile retail operations (Wu, Yang, & Wei, 2004), and also to obtain fuzzy if-then rules (Wu, Yang, & Hung, 2005). Debels et al. (2006) integrated a scatter search with EM for the solution of resource constraint project scheduling problems. Their

research is the first paper that includes an EM type methodology for the combinatorial optimization problem. The result of their experiments show that although Electromagnetism designed to solve problem with limited variables, the algorithm can be extended to solving combinatorial problem and the hybrid method of incorporating EM type analysis outperforms the current best solution available in the literature.

When we extend the EM algorithm to combinatorial optimization problems, the first important step is the representation of a solution. Bean (1994) introduced a random-key (RK) to make sequence of jobs based on the generate point from feasible region.

2.3 Minimizing Earliness and Tardiness Penalties Problem

According to Gandibleux and friends (2003) scheduling is the arrangement of entities (people, tasks, vehicles, lectures, exams, meeting, etc) in to pattern in space time in such a way that constrain are satisfied and certain goals are achieved. Constructing a schedule is in which time, space and other (often limited) resources have to be considered in the arrangement.

The constraints are relationships among the entities or between entities and the pattern that limit the construction of earliness and tardiness. Constraint can be classified as hard or soft. Hard constraints must not be violated, a penalty is applied and the solution is still considered to be feasible. In practice, the scheduling activity can be regarded as a search problem for which it is required to find any feasible schedule or as an optimization problem for which the best feasible schedule is sought. Scheduling Machine consists of two kinds of types: single machine scheduling and parallel machine scheduling, single machine scheduling where only consisting of one machine that are used sequentially in a series of work while the parallel process of scheduling a machine composed of several machines that are used together in one time in a work process.

Minimizing earliness and tardiness penalties problem have many different of type too (generally due to different input data and objective functions). One of the simplest to state, but not easy to solve at all, is the problem of sequencing n jobs, given its processing time and due dates (distinct for each job), and with the objective function being to minimize the total tardiness.

The Minimizing earliness and tardiness penalties problem addressed can be extended by the inclusion of precedence constraint for jobs, ready times, resources limitation, etc. From an objective function's point of view, we may want to minimize the makespan, the total tardiness, the mean tardiness, the number of tardy jobs or even a combination of these objectives which would characterize a multi criteria problem. Refer to Gen and Cheng (2007) it is easy to see the large variety of problems that we may face in practice. Even looking at very complex industrial manufacturing systems, it is not hard to find situation in which a simple Minimizing earliness and tardiness penalties problem should be solved (Ow and Morton, 1989).

There are many research has done by scientists about Minimizing earliness and tardiness penalties problem, among other are Ragatz (1993) a branch and bound (B&B) method is proposed but only small instances could be solved to optimality, Raman et al (1997) and Lee et al (1997) use dispatch rules based on the calculation of priority index to build an approximate schedule, which is then improved by the application of a local search procedure. The ACTS heuristic proposed by Lee et al (1997) had an impressive performance for this problem, considering its simplicity. The scientists also using metaheuristics to do their research for example Rubin and Ragatz (1995) a new crossover operator was developed and a Genetic Algorithm (GA) was applied to a set of tests problems. The results obtained by the GA approach were compared with the ones from a B&B and with a multiple start (MS) algorithm and they conclude that MS outperformed the B&B and the GA in many instances, considering running time and quality of solutions as performance measure. Of course, the instances in which MS outperformed B&B were the ones where the exact method did not find an optimal solution before a limit on the number of nodes was reached. The B&B was truncated in such cases, returning sub optimal schedules. Given these results Tand and Narasimhan (1997) chose the MS technique as a baseline benchmark for conducting comparisons with the simulated annealing (SA) approach they proposed. Their conclusion was that SA outperformed MS in all but three instances, with percentage improvements not greater than 6%. More recently, Franca et al (2001) proposed a new memetic algorithm (MA) that outperformed the previous approaches.

In our project, we only use an Electromagnetism-like mechanism algorithm for the solution of resource-constrained Minimizing earliness and tardiness penalties problems. The random-key approach to represent a schedule incorporated with the EM methodology are applied to solve a Minimizing earliness and tardiness penalties problem and the objective is to minimize the total sum of earliness and tardiness penalties. A detailed formulation of the problem is described as follows: a set of n independent jobs (J_1, J_2, \dots, J_n) with positive processing time p_j has to be scheduled without preemptions on a single machine that can handle at most one job at a time and should and has a due date d_j . The machine is assumed to be continuously available from time zero onwards and unforced machine idle time is not allowed. For any given schedule, the earliness and tardiness of J_j can be, respectively, defined as $E_j = \max(0, d_j - C_j)$ and $T_j = \max(0, C_j - d_j)$, where C_j is the completion time of J_j . The objective is then to find a schedule that minimizes the sum of the earliness and tardiness penalties of all jobs $\sum_{j=1}^n (\alpha_j E_j + \beta_j T_j)$, where α_j and β_j are the earliness and tardiness penalties of job J_j . The inclusion of both earliness and tardiness costs in the objective function is compatible with the philosophy of just-in-time production, which emphasizes producing goods only when they are needed. The early cost may represent the cost of completing a product early, the deterioration cost for a perishable goods or a holding (stock) cost for finished goods. The tardy cost can represent rush shipping costs, lost sales and loss of goodwill. It is assumed that no unforced machine idle time is allowed, so the machine is only idle if no job is currently available for processing.

Finally, to find a solution we try to implement electromagnetism-like mechanism algorithm for Minimizing earliness and tardiness penalties problem solving by using language C++ as tool of programming.

2.4 C++ Programming Language

C++ (pronounced "see plus plus") is a general-purpose computer programming language. It is a statically typed free-form multi-paradigm language supporting procedural programming, data abstraction, object-oriented programming, and generic programming. Since the 1990s, C++ has been one of the most popular commercial programming languages.

Bell Labs' Bjarne Stroustrup developed C++ (originally named "C with Classes") in 1983 as an enhancement to the C programming language. Enhancements started with the addition of classes, followed by, among many features, virtual functions, operator overloading, multiple inheritance, templates, and exception handling. The C++ programming language standard was ratified in 1998 as ISO/IEC 14882:1998, the current version of which is the 2003 version, ISO/IEC 14882:2003. A new version of the standard (known informally as C++0x) is being developed.

In 1983, the name of the language was changed from C with Classes to C++. New features that were added to the language included virtual functions, function name and operator overloading, references, constants, user-controlled free-store memory control, improved type checking, and new comment style (`//`). In 1985, the first edition of *The C++ Programming Language* was released, providing an important reference to the language, as there was not yet an official standard. In 1989, Release 2.0 of C++ was released. New features included multiple inheritance, abstract classes, static member functions, const member functions, and protected members. In 1990, *The Annotated C++ Reference Manual* was released and provided the basis for the future standard. Late addition of features included templates, exceptions, namespaces, new casts, and a Boolean type.

As the C++ language evolved, a standard library also evolved with it. The first addition to the C++ standard library was the stream I/O library which provided facilities to replace the traditional C functions such as `printf` and `scanf`. Later, among the most significant additions to the standard library, was the Standard Template Library.

Many forms of increased support of generic programming in C++ have been proposed since the initial template design. In particular, constraints based on variations of inheritance have been proposed but not adopted as they tend to focus on only part of the problem and to rely on non-scalable mechanisms. The design, implementation, and use of the Standard Template Library have contributed significantly to the appreciation of templates and generic programming techniques. Most recent proposals focus on the notion of concept, which can roughly be summarized as a type system for template arguments. Currently, two main approaches to the design of concepts are being debated in the C++ community and standards committee.

CHAPTER III

RESOLUTION METHODE

3.1 Elektromagnetism-like Mechanism (EM) Heuristics Algorithm

3.1.1 General Scheme

Algorithm Methaheuristics Electromagnetism-like mechanism (EM) consists of four phases. These are :

- a. Initialization of algorithm
- b. Local Search
- c. Calculation of the total force exerted on each particle
- d. Movement along the direction of the force.

Table 3.1 Main Algorithm of EM

ALGORITHM 1. $EM(m, MAXITER, LSITER, \delta)$

m : number of sample points

$MAXITER$: maximum number of iterations

$LSITER$: maximum number of local search iterations

δ : local search parameter, $\delta \in [0, 1]$

1: Initialize ()

2: Iteration $\leftarrow 1$

3: **while** iteration < $MAXITER$ **do**

4: Local ($LSITER, \delta$)

5: $F \leftarrow CalcF ()$

6: Move (F)

7: iteration $\leftarrow + 1$

8: **end while**

3.1.2 Initialization

The procedure Initialize is used to sample m points randomly from the feasible domain, which is an n dimensional hyper-cube. Each coordinate of a point is assumed to be uniformly distributed between the corresponding upper bound and lower bound. After a point is sampled from the space, the objective function value for the point is calculated using the function pointer $f(x)$ (Algorithm 2, line 6). The procedure ends with m points identified, and the point that has the best function value is stored in x^{best} (line 8).

Table 3.2 Algorithm Initialize

ALGORITHM 2. Initialize()

```
1: for  $i = 1$  to  $m$  do
2:   for  $k = 1$  to  $n$  do
3:      $\lambda \leftarrow U(0, 1)$ 
4:      $x_k^i \leftarrow lk + \lambda(uk - lk)$ 
5:   end for
6:   Calculate  $f(x^i)$ 
7: end for
8:  $x^{\text{best}} \leftarrow \operatorname{argmin}\{f(x^i), \forall i\}$ 
```

3.1.3 Local Search

The procedure Local is used to gather the local information for a point x^i . The parameters, LSITER and δ that are passed to this procedure, represent the number of iterations and the multiplier for the neighborhood search, respectively. The procedure iterates as follows: First, the maximum feasible step length (Length) is calculated according to the parameter δ (Algorithm 3, line 2). Second, for a given i , improvement in x^i is sought coordinate by coordinate (lines 5–13). For a given coordinate, the point x^i is assigned to a temporary point y to store the initial information. Next, a random number is selected as a step length and the point y is moved along that direction. If the point y observes a better point within LSITER iterations, the point x^i is replaced by y and the neighborhood search for

point i ends (lines 14–17). Finally the current best point is updated (line 22). This is a simple random line search algorithm applied coordinate by coordinate. This procedure does not require any gradient information to perform the local search. Instead of using other powerful local search methods (Solis and Wets, 1981), we have utilized this procedure because we wanted to show that even with this trivial method, the algorithm shows promising convergence properties.

Table 3.3 Algorithm Local Search

ALGORITHM 3. Local(LSITER, δ)

```

1: counter  $\leftarrow$  1
2: Length  $\leftarrow$   $\delta(\max_k\{u_k - l_k\})$ 
3: for  $i = 1$  to  $m$  do
4:   for  $k = 1$  to  $n$  do
5:      $\lambda_1 \leftarrow U(0, 1)$ 
6:     while counter  $<$  LSITER do
7:        $y \leftarrow x^i$ 
8:        $\lambda_2 \leftarrow U(0, 1)$ 
9:       if  $\lambda_1 > 0.5$  then
10:         $y_k \leftarrow y_k + \lambda_2(\text{Length})$ 
11:       else
12:         $y_k \leftarrow y_k - \lambda_2(\text{Length})$ 
13:       end if
14:       if  $f(y) < f(x^i)$  then
15:         $x^i \leftarrow y$ 
16:        counter  $\leftarrow$  LSITER - 1
17:       end if
18:       counter  $\leftarrow$  counter + 1
19:     end while
20:   end for
21: end for
22:  $x^{\text{best}} \leftarrow \operatorname{argmin}\{f(x^i), \forall i\}$ 

```

3.1.4 Calculation Force

The *superposition principle* (Figure 3.1) of electromagnetism theory states that the force exerted on a point via other points is inversely proportional to the distance between the points and directly proportional to the product of their charges (Cowan,1968).

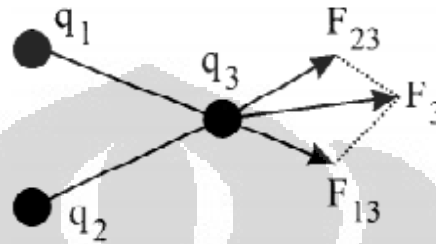


Figure 3.1 Illustration of Coulomb's law.

In each iteration we compute the charges of the points according to their objective function values. However, in our heuristic the charge of each point is not constant and changes from iteration to iteration. The charge of each point i , q^i , determines point i 's power of attraction or repulsion. This charge is evaluated as,

$$q^i = \exp \left[-n \frac{f(x^i) - f(x^{best})}{\sum_{k=1}^m (f(x^k) - f(x^{best}))} \right], \forall i \quad (1)$$

In this way, points that have better objective values possess higher charges. We multiply the fraction by the dimension n , because in higher dimensions the number of points in the population tends to get large. As a result of this, the fraction may become very small, and may cause overflow problems in calculating the exponential function. We define the charge, q^i according to the relative efficiency of the objective function value of the corresponding point in the population. This is clearly not the unique nor the optimal choice for this calculation. An alternative calculation, which rank the points according to their objective function values, may be used here. Our experiments have shown that the proposed calculation in Eq. (1) is satisfactory for our study.

Notice that, unlike electrical charges, no signs are attached to the charge of an individual point in Eq. (1). Instead, we decide the direction of a particular force between two points after comparing their objective function values. Hence, the total force F^i exerted on point i is computed by the following equation:

$$F^i = \sum_{j \neq i}^m \left(\begin{array}{l} (x^j - x^i) \frac{q^i q^j}{\|x^j - x^i\|^2} \text{ if } f(x^j) < f(x^i) \\ (x^j - x^i) \frac{q^i q^j}{\|x^j - x^i\|^2} \text{ if } f(x^j) > f(x^i) \end{array} \right) \quad (2)$$

As seen in Algorithm 4 (lines 7–8), between two points, the point that has a better objective function value attracts the other one. Contrarily, the point with worse objective function value repels the other (lines 9–10). Since x^{best} has the minimum objective function value, it acts as an absolute point of attraction, i.e., it attracts all other points in the population. When we examine the algorithm carefully, we can see that the determination of a direction via total force vector resembles the statistical estimation of the gradient of f . However, the estimation computed by our heuristic is different since this direction depends on the Euclidean distance between the points. So, the points that become close enough may lead each other to a direction other than the statistically estimated one.

Table 3.4 Algorithm Calculation Force

ALGORITHM 4. CalcF()

- 1: **for** $i = 1$ to m do
- 2: $q^i \leftarrow \exp \left[-n \frac{f(x^i) - f(x^{\text{best}})}{\sum_{k=1}^m (f(x^k) - f(x^{\text{best}}))} \right]$
- 3: $F^i \leftarrow 0$
- 4: **end for**
- 5: **for** $i = 1$ to m do
- 6: **for** $j = 1$ to m do
- 7: **if** $f(x^j) < f(x^i)$ then
- 8: $F^i \leftarrow F^i + (x^j - x^i) \frac{q^i q^j}{\|x^j - x^i\|^2}$ {Attraction}
- 9: **else**
- 10: $F^i \leftarrow F^i - (x^j - x^i) \frac{q^i q^j}{\|x^j - x^i\|^2}$ {Repulsion}
- 11: **end if**
- 12: **end for**
- 13: **end for**

3.1.5 Movement

After evaluating the total force vector F^i , the point i is moved in the direction of the force by a random step length as given in Eq. (3). Here two parameters have to be defined: one is the random step length, λ , is assumed to be uniformly distributed between 0 and 1. Obviously, there are many other distributions that can be used in calculation of this step length. But for ease of computation, we have applied uniform distribution. We have selected the step length randomly in order to ensure that the points have a nonzero probability to move to the unvisited regions along this direction. In our future work, we hope to show the convergence in probability by using this random step length. The other is RNG is a vector whose components denote the allowed feasible movement toward the upper bound, uk , or the lower bound, lk , for the corresponding dimension (Algorithm 5, lines 6–10). Furthermore, the force exerted on each particle is normalized so that we can maintain the feasibility. Thus,

$$x^i = x^i + \lambda \frac{F^i}{\|F^i\|} \text{ (RNG)} \quad i = 1, 2, \dots, m \quad (3)$$

Algorithm 5 gives the pseudo-code of the Move procedure. Note that the best point, x_{best} , is not moved and is carried to the subsequent iterations (line 2). This suggests that we may avoid the calculation of the total force on the current best point in Algorithm 4 (yet the computational effort for calculating the total force on current best point is negligible).

Table 3.5 Algorithm Move

ALGORITHM 5. Move(F)

- 1: **for** $i = 1$ to m **do**
- 2: **if** $i \neq \text{best}$ **then**
- 3: $\lambda \leftarrow U(0, 1)$
- 4: $F^i \leftarrow \frac{F^i}{\|F^i\|}$
- 5: **for** $k = 1$ to n **do**
- 6: **if** $F_{k}^i > 0$ **then**

```

7:       $x_k^i \leftarrow x_k^i + \lambda F_k^i (u_k - x_k^i)$ 
8:      else
9:       $x_k^i \leftarrow x_k^i + \lambda F_k^i (x_k^i - u_k)$ 
10:     end if
11:  end for

```

3.2 EM for Minimizing Earliness and Tardiness Penalties Problem (METPP)

In this project we added one more method, that is random key-technique to make EM enables to solve this kind of problem. In this project some procedures like local search, particle charge and electrostatic force are modified because the time complexity is high and to obtain a better solution quality.

Random key method is an additional step that is used to enable EM to solve scheduling problems. Random key technique has a simple concept and can be applied easily. When we obtain a k -dimension solution, we sort the value corresponding to each dimension, any sorting algorithm can be used and in this project we use quick sort because its time complexity is $O(n \log n)$. After having a sequence, we can use it to compute the objective function value of this sequence.

Figure 3.2 demonstrates a 10-dimension solution. Values of dimension 1, 2 and 3 are 0.5, 9.6 and 3.0. The rest are shown in figure 3.2. Then, we apply the random-key method to sort these values in ascending order. Thus sequence at position 1 is 8 that means we schedule job 8 in the beginning and job 2 is scheduled at the last position. By random-key method, the continuous electromagnetism algorithm is able to be applied to solve different kinds of sequencing problems.

		Activities									
		1	2	3	4	5	6	7	8	9	10
Before		0.5	9.6	3.0	2.9	2.2	8.0	4.2	0.1	7.1	5.6
(a) Value of Activities											
After		8	1	5	4	3	7	10	9	6	2
(b) Schedule List											

Figure 3.2. An Example of Random Key Method

3.2.1 Main Algorithm

Generally, Main Algorithm For Minimizing Earliness and Tardiness Penalties Problem not different with EM Main Algorithm for Global optimization as shown on Table 3.5 below.

Table 3.6 EM Main Algorithm For METPP

The Algorithm 1. EM()

1. initialize
 2. **while** (hasn't met stop criterion) **do**
 3. local search ()
 4. calculate total force $F()$
 5. move particle by $F()$
 6. evaluate particles()
 7. **End While**
-

3.2.2 Initialization

The algorithm 2 initiates particles in the population. The initial value is between lower bound and upper bound. The lower bound and upper bound are set between (-1,1). After all particles are generated, the Random key method is used to generate sequence of the corresponding values of of each particle. As soon as we obtain the sequence, the objective values of these particles are evaluated and

we can obtain current best solution from these solutions (algorithm 2, line 8 and 9).

Table 3.7 EM Initialize Algorithm For METPP

Algorithm 2. Initialize()

```

1: for  $i = 1$  to  $m$  do
2:   for  $k = 1$  to  $n$  do
3:      $\lambda \leftarrow U(0, 1)$ 
4:      $x_k^i \leftarrow l_k + \lambda(u_k - l_k)$ 
5:   end for
6: end for
7: find sequence by random-key method
8: evaluate particles ()
9:  $x^{best} \leftarrow \operatorname{argmin}\{f(x_i), \forall i\}$ 

```

3.2.3 Local Procedure

The algorithm that perturbs each dimension of the best solution (Algorithm 3, line 5–12) then finds its corresponding sequence and their objective value. This new solution will replace the worst solution when its objective value is better than the worst solution (Algorithm 3, line 12–15). Therefore, it attempts to improve average solution quality iteratively for LSITER times. This procedure may find a better solution to substitute the current best solution (Algorithm 3, line 16–20).

Table 3.8 EM Local Search Algorithm For METPP

The Algorithm 3. Local (LSITER)

```

1:  $\text{Length} \leftarrow \operatorname{argmax}\{u_k - l_k, \forall k\}$ 
2:  $i \leftarrow \operatorname{argmin}\{f(x^i), \forall i\}$ 
3: for  $j = 1$  to LSITER do
4:    $y \leftarrow x^i$ 
5:   for  $k = 1$  to  $n$  do
6:      $\lambda \leftarrow U(0, 1)$ 
7:     if  $U(0, 1) > 0.5$  then
8:        $y_k \leftarrow y_k + \lambda(\text{Length})$ 
9:     else
10:       $y_k \leftarrow y_k - \lambda(\text{Length})$ 

```

```

11:     end if
12:   end for
13:    $l = \operatorname{argmax} \{f(x^i), \forall_i\}$ 
14:   if  $f(y) < f(x^l)$  then
15:      $x^l \leftarrow y$ 
16:   end if
17:   if  $f(y) < f(x^i)$  then
18:      $x^{best} \leftarrow y$ 
19:      $x^i \leftarrow y$ 
20:      $i \leftarrow l$ 
21:   end if
22: end for

```

3.2.4 Particle charges, electrostatic force and move

The study uses the total force algorithm proposed by Debels et al. (2006), which determines the force exerted on particle i by point j that does not use the fixed charge of q_i and q_j . Instead, q_{ij} depends on the relative deviation of $f(x^i)$ and $f(x^j)$. Thus this particle charge is calculated as follows:

$$q_{ij} = \frac{F(x^i) - f(x^j)}{F(x^{worst}) - f(x^{best})} \quad (3)$$

If the objective value $f(x^i)$ is larger than $f(x^j)$, particle j will attract particle i . On the other hand, when $f(x^i) < f(x^j)$, a repulsion effect is occurred. There is no action when $f(x^i) = f(x^j)$ because q_{ij} is equal to zero. After the q_{ij} is obtained, the force on particle i by particle j is

$$F^{ij} = (x^j - x^i) \cdot q_{ij} \quad (4)$$

Thus the particle x^i moves to $x^i + F^{ij}$ in the direction of particle x^j . This method is similar to the path relinking method by Glover, Laguna, and Marti (2000) which gradually moves from one point to another by Debels et al.(2006).

Table 3.9 EM Movement Algorithm For METPP

Algorithm 4. CalcF and move (x^i)

```

1:  $F^i \leftarrow 0$ 
2: for  $j = 1$  to  $m$  do
3:   if  $x^i \neq x^j$  then

```

$$f(x^i) - f(x^j)$$

```

4:       $q_{ij} \leftarrow \frac{1}{f(x^{\text{worst}}) - f(x^{\text{best}})}$ 
5:  $F^{ij} \leftarrow (x^j - x^i)q_{ij}$ 
6:       $x^i \leftarrow x^i + F^{ij}$ 
7: end if
8: end for

```

Finally, in order to maintain the feasibility of each solution, we check the boundary feasibility by the Algorithm 5.

Table 3.10 EM Check Boundary Algorithm For METPP

Algorithm 5. Check Boundary()

```

1: for  $i = 1$  to  $m$  do
2:   for  $j = 1$  to  $n$  do
3:     if  $x^{ij} > u_j$  then
4:        $x^{ij} \leftarrow u_j$ 
5:     else if  $x^{ij} < l_j$  then
6:        $x^{ij} \leftarrow l_j$ 
7:     end if
8:   end for
9: end for

```

CHAPTER IV

IMPLEMENTATION

In this chapter, we use the modified Electromagnetism-like Mechanism algorithm to solve the Minimizing Earliness and Tardiness Penalties Problem and analyze the efficiency of it using C++ Language and performed computational test on Intel Pentium-4 1.4 GHz Note book. First, we prepare installation DEV C++ in my computer. Later, we make coding regarding single machine problem with reference Electromagnetism-like Mechanism algorithm that consists of four phases, these are initialization, local search, calculate Force, and movement. Finally, results of sequential job with minimum function objective are finding.

4.1 Initialization Procedure

In this section we determine m population as initial number of sequential consists of n job j ($j_1, j_2, j_3, \dots, j_n$), job 1 to n sample points randomly from the feasible domain, each coordinate of a point is assumed to be uniformly distributed the corresponding upper bound and lower bound, after n sample points are sampled from the space sequential job can defined as sorting ascending of sample point.

After we have sequential of job n , we determine due date each job (d_j) and processing time each job (p_j), the machine is assumed to be continuously available from time zero onwards and unforced machine idle time is not allowed. Job j becomes available for processing at the beginning, requires a processing time p_j and should be completed on its due date d_j , so, completion time each job C_j can find and the earliness and tardiness of job j can be. And this process re-continued until m population.

Finally we can calculate the sum of earliness and tardiness and multiple by penalty represented by value of α and β that defined in variable global and it is value of function objective, and we can find a schedule that minimum function objective in sequential job initialization.

To make sequential process of job j , the random key approaches from the feasible domain corresponding lower bound and upper bound are applied firstly (Table 4.1 line 1-

3), and x defined as lower bound + lamda (upper bound – lower bound) like procedure shown in table 4.1 line 4-6.

Table 4.1 Establishing a Random Key with C++ Pseudo Code

```

1. double lamda;
2. srand((unsigned)time(NULL));
3. lamda=((double) rand() / (RAND_MAX+1)) ;
4. for (int i=1;i<=m;++i)
5.     for (int j=1;j<=n;++j)
6.         x[i][j]=low+(lamda*(up-low));

```

After random key are generated, we make sequencing using quick sort ascending of n sample point that presenting sequencing process of job j as algorithm shown in Table 4.2.

Table 4.2 Sorting Ascending Pseudo Code

```

1. for (int i = 1; i <= n; i++)
2. for(int j = 1; j <=n - 1; j++)
3.     if (x[i][j] > x[i][j + 1])
4.         double Temp = x[i][j];
5.         x[i][j] = x[i][j + 1];
6.         x[i][j + 1] = Temp;
7. for (int ib=1;ib<=n;ib++)
8.     hasil[ib]=x[i][ib];
9. for (int ic=1;ic<=n;ic++)
10. for (int ja=1;ja<=n;ja++)
11.     if (awal[ic]==hasil[ja])
12.         indeks[ja]=ic;

```

Next step is calculate earliness and tardiness and sum of tardiness earliness penalty as function objective, for calculating them the calculation of due date d_j and calculation completion time C_j must defined firstly (Table 4.3 line 1-12), and sum of earliness and tardiness defined as $z = \text{alfa} * \text{esum} + \text{beta} * \text{tsum}$ can calculated as line 13-21 Table 4.3 below:

Table 4.3 Calculate Earliness, Tardiness and Function Objective Pseudo Code

```

1. for (int id=1;id<=n;id++)
2.     dd[id]=d[indeks[id]];
3.     pt[id]=p[indeks[id]];
4.     ct[1]=pt[1];
5. for (int id=2;id<=n;id++)
6.     ct[id]=ct[id-1]+pt[id];
7.     if (dd[id]>=ct[id])
8.         e[id]=dd[id]-ct[id];
9.         t[id]=0;
10.    if(dd[id]<ct[id])
11.        e[id]=0;
12.        t[id]=ct[id]-dd[id];
13. double tempsum=0;
14. double tempsum2=0;
15. for (int j=1;j<=n;j++)
16.     tempsum=tempsum+e[j];
17.     tempsum2=tempsum2+t[j];
18. for (int i=1;i<=n;i++)
19.     esum[i]=tempsum;
20.     tsum[i]=tempsum2;
21.     z[i]= alfa*esum[i]+beta*tsum[i];

```

After function objective z was fined for each schedule i ($i_1, i_2, i_3, \dots, i_m$) we make algorithm code to finding function objective minimum and stored as z^{best} using algorithm showing on Table 4.4 below:

Table 4.4 Searching Minimum Function Objective Value Pseudo Code

```

1. for (int jz=1;jz<=m;++jz)
2.     min_value[1]=z[1];
3. for (int jz=1;jz<=m;++jz)
4.     if (z[jz]<=min_value[1])
5.         min_value[1]=z[jz];
6.         indexbestmin=jz;

```

4.2 Local Search Procedure

In this section we determine *local* procedure to gather the local information for a point x^i , the parameter *lsiter* is passed to this procedure represent the number of iteration.

The algorithm that perturbs each dimension of the best solution then finds its corresponding sequence and their objective value. This new solution will replace the worst solution when its objective value is better than the worst solution, therefore, its attempts to improve solution quality iteratively for *lsiter* times, this procedure may find a better solution to substitute the current best solution.

Beginning Local Search implementation, length is defining with upper bound minus lower bound (up-low) and y as a new point taken from x^{best} initialization, process before local search. After y defined as x^{best} defined, the random key method are generated as explained at Table 4.1 line 1-3. So, y_j defined as $y_j + (\text{lamda} * \text{length})$ if $\text{lamda} > 0.5$ and if $\text{lamda} \leq 0.5$ y_j defined as $y_j - (\text{lamda} * \text{length})$ like procedure showing in Table 4.5 line 10-13.

Table 4.5 Defining y_j in Local Search Procedure Pseudo Code

```
1. int counter=1;
2. double length=up-low;
3. double lamda;
4. srand((unsigned)time(NULL));
5. while (counter<lsiter)
6.   for (int i=1;i<=1;++i)
7.     for (int k=1;k<=n;++k)
8.       y[i][k]=x[indexbestmin][k];
9.       lamda=((double) rand() / (RAND_MAX+1)) ;
10.      if (lamda>0.5)
11.        y[i][k]=y[i][k]+(lamda*length);
12.      else
13.        y[i][k]=y[i][k]-(lamda*length);
14. if (zz[i]<= min_value[1])
15.   counter=counter+lsiter;
16.   for (int k=1; k<=n; k++)
17.     yy[k]=y[i][k]
18. end if
19. counter=counter+1;
```

Next step, after y_j was defined, sequential of job y can be made by sorting ascending of y_j and calculate earliness and tardiness and sum of tardiness earliness penalty as function objective in local search can be obtained using same way in initialization.

The objective of local search is to get the value of z (function objective) better than z^{best} , in case z value in local search not yet better than z^{best} , search was repeated by counter = counter + 1 and running will stop when z is better, maximum iteration is value of $lsister$ was defined. Finally, we can store y_j from $z^{newbest}$.

4.3 Calculate Force Procedure

The implementation uses the total force algorithm, the force exerted on particle i by point j that does not use the fixed charge of q_i and q_j . Instead, q_{ij} depends on the relative deviation of $f(x^i)$ and $f(x^j)$. Thus this particle charge is calculated as $q_{ij} = f(x^i) - f(x^j) / f(x^{worst}) - f(x^{best})$

If the objective value $f(x^i)$ is larger than $f(x^j)$, particle j will attract particle i . On the other hand, when $f(x^i) < f(x^j)$, a repulsion effect is occurred. There is no action when $f(x^i) = f(x^j)$ because q_{ij} is equal to zero. After the q_{ij} is obtained, the force on particle i by particle j is $F^{ij} = (x^j - x^i) \cdot q_{ij}$.

Table 4.6 Calculating Charge and Force Pseudo Code

```

1. double F[5]=0;
2. for (int j=1; j<=m; j++)
3.   for (int k=1; k<=n; k++)
4.     if (x[j][k]!=y[j][k])
5.       q[j][k]= (z[j])-(zz[1]) / (z[indexbestmax])-(z[indexbestmin]);
6.       F[j][k]= (y[j][k]-x[j][k])*q[j][k];
7.     end if
8.   end for
9. end for

```

4.4 Movement Procedure

The Implementation of movement is to find the direction of point x^i , magnitude and direction of x^i defined as $x^i = x^i + F^{ij}$, that F^{ij} defined as procedure before it. With other sentences can be explained that the particle x^i move to $x^i + F^{ij}$.

Table 4.7 Movement x^i Pseudo Code

```
1. for (int i=1;i<=m;++i)
2.   for (int k=1;k<=n;++k)
3.     xmove[i][k]=x[i][k]+F[i][k];
4.   end for
5. end for
```

4.5 Main Program

Finally, in order to running the procedure initialization, local search, calculation Force and movement, we make the main program in the last of C++ frame work as follows:

Table 4.8 Main Program Pseudo Code

```
void EM (int maxiter, int lsiter, int iter, int sched)
1. Begin
2. inisialisasi();
3. int iteration=1;
4. while (iteration<maxiter)
5.   localsearch(lsiter);
6.   calculF();
7.   move();
8.   iteration=iteration+1;
9. end
```

4.6 Program Set-Up

In the program set-up, some important parameters which are discussed in the general scheme of the original Electromagnetism-Like Mechanism algorithm are used. The parameters are defined as follow:

m = number of sample

3

n = number of job	5
maxiter : maximum number of iteration	10
lsiter : maximum number of local search iteration	10

So, in this study with theme Minimizing Earliness and Tardiness Penalties Problem same important parameter are used, The parameters are defined as variable global in C++ code and we can change the parameter as we would like. Showing in Table 4.8 is example of program setup for 5 jobs.

Table 4.9 Example Program set-up for 5 jobs

```

1. int m=2;
2. int n=5;
3. double low= -1;
4. double up= 4;
5. double lamda;
6. double x[3][6],awal[6],hasil[6];
7. int indeks[6];
8. double q[3][6];
9. double f[3][6];
10. double xmove[3][6];
11. double d[6]= {0,9,14,12,8,13};
12. double p[6]= {0,6,10,11,7,12};
13. double dd[6],pt[6],ct[6];
14. double e[6],t[6], esum[6], tsum[6];
15. double z[3];
16. int alfa=1;
17. int beta=1;
18. double y[2][6];//=x for localsearch
19. double yy[6];
20. double zz[2];//=Z for localsearch
21. double zzz[3];//=Z for move

```

CHAPTER V

EXPERIMENTAL STUDY

This study proposed a Electromagnetism-like Mechanism (EM) metaheuristics that modified EM meta-heuristics in solving the Minimizing Earliness and Tardiness Penalties Problem to minimizing the earliness and tardiness penalty. In order to evaluate the performance of this framework, the output is compared with EDD, SPT and LPT. Across these experiments, we adopt the scheduling instances of job size are 5, 10, 15, 20. Each scheduling instances of job size, the function objective is to minimizing the earliness and tardiness penalty, so, we set 1 for earliness (alfa) penalty and 1 for tardiness (beta) penalty.

The EDD (Earliest Due Date) scheduling method is scheduling with jobs sequenced according to their due dates, the SPT (Shortest Processing Time) method is scheduling method Jobs with the shortest processing time are scheduled first, and the LPT (Largest Processing Time) method scheduling method Jobs with the largest processing time are scheduled first.

5.1 Data Entry

For scheduling instances of job size are 5, we have data entry are due date (d_j) and processing time (p_j) as shown in Table 5.1, earliness data entry (alfa) is 1 and tardiness data entry is 1. This data entry we have set in program set-up at variable global of C++ code, procedure setting of entry data and example have explained in subsection program setup chapter IV.

Table 5.1 Data Entry for 5 jobs

Job i	1	2	3	4	5
d_i	9	14	12	8	13
p_i	6	10	11	7	12

For scheduling instances of job size are 10, we have data entry are due date (d_j) and processing time (p_j) as shown in Table 5.2, earliness data entry (alfa) is 1 and tardiness data entry is 1. This data entry we have set in program set-up at variable global of C++ code.

Table 5.2 Data Entry for 10 jobs

Job i	1	2	3	4	5	6	7	8	9	10
d_i	10	15	14	8	16	9	5	12	11	7
p_i	6	13	14	7	12	8	4	10	9	5

For scheduling instances of job size are 15, we have data entry are due date (d_j) and processing time (p_j) as shown in Table 5.3, earliness data entry (alfa) is 1 and tardiness data entry is 1. This data entry we have set in program set-up at variable global of C++ code.

Table 5.3 Data Entry for 15 jobs

Job i	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
d_i	12	11	17	8	16	6	16	12	11	13	18	15	17	14	7
p_i	3	5	14	7	12	4	15	10	10	11	13	9	4	6	6

For scheduling instances of job size are 20, we have data entry are due date (d_j) and processing time (p_j) as shown in Table 5.4, earliness data entry (alfa) is 1 and tardiness data entry is 1. This data entry we have set in program set-up at variable global of C++ code.

Table 5.4 Data Entry for 20 jobs

Job i	1	2	3	4	5	6	7	8	9	10
d_i	16	11	17	8	13	6	20	12	11	13
p_i	3	6	5	7	12	4	18	10	9	11
Job i	11	12	13	14	15	16	17	18	19	20
d_i	18	15	17	14	7	5	19	22	18	21
p_i	16	9	4	13	6	2	15	17	13	19

5.2 Result and Comparison

The result of the Electromagnetism-like Mechanism (EM) Algorithm for the Minimizing Earliness and Tardiness Penalties Problem for each instances jobs and data entry as explained in subsection before, indicates the best solution obtained by running trial of application of 20 times running, the result are obtained from setting up other parameter as follow:

m: number of initialization schedule = 3

local search iteration = 10

maximum iteration = 10

The best result for 5 jobs are by EM = 69, by EDD=72, by SPT=72, by LPT=100. The best result for 10 jobs are by EM = 278, by EDD=72, by SPT=72, by LPT=100. The best result for 15 jobs are by EM = 621, by EDD=703, by SPT=626, by LPT=1084. The best result for 20 jobs are by EM = 1242, by EDD=1383, by SPT=1251, by LPT=2402. Complete calculation for EDD, SPT, LPT method for each job instances are shown in appendix 1, appendix 2, appendix 3 and appendix 4 in the last section of this paper.

Table 5.5 The comparison result between EM, EDD, SPT and LPT

Instances	EM	EDD	SPT	LPT
5 jobs	69	72	72	100
10 jobs	278	293	286	470
15 jobs	621	703	626	1084
20 jobs	1242	1383	1251	2402

CHAPTER VI

CONCLUSION AND PERSPECTIVE

6.1 Conclusion

End of this paper, we can obtain conclusion regarding implementation of Electromagnetism-like Mechanism to Minimizing Earliness and Tardiness Penalties Problem as follow:

1. Owing the development of a random key method, the EM algorithm is able to be applied in solving the sequence problem.
2. The random key method to represent a schedule and incorporate the EM methodology to solve the Minimizing Earliness and Tardiness Penalties Problem are easy to implement, but it have disadvantage if there are two particles with have same value. therefore, we must apply the best sorting algorithm.
3. Essential of EM Algorithm is attract and repulsion particle and in this case particle j will attract particle i if function objective value $f(x^i)$ is larger then $f(x^j)$, on the other hand, when $f(x^i) < f(x^j)$ a repulsion effect is occurred.
4. According to experimental result, for better quality of solution, we can set population, iteration and local search iteration higher for scheduling high number of job instances.

6.2 Perspectives

1. To improve the performance of the EM algorithm for Minimizing Earliness and Tardiness Penalties Problem, a hybrid method can be applied.
2. For future, The EM algorithm, we will try to apply for solving parallel machine scheduling problem.
3. For better quality interface, we can use visual programming language.

REFERENCES

- Bean, J.C., *Genetic algorithms and random keys for sequencing and optimization* : ORSA Journal on computing (1994)
- Birbil, S.I. and Fang,S.C, *An electromagnetism-like mechanism for global optimisation* : Journal of global optimisation (2003)
- Birbil, S.I.,Gukan G., and Listes O., *Solving Stochastic mathematical programs with complementarity constarin using simulation* : Winter simulation conference (2004)
- Cowan, E. W. *Basic Electromagnetism*, Academic Press, New York (1968)
- D.S. Malik, *C++ Programming: Program Design Including Data Structures, 5th International Edition*, Course Technology Creighton University (2011)
- Debels,D., Reyck,B.D,Leus, R., and Vanhoucke,M., *A hybrid scatter search/electromagnetism meta heuristic for project scheduling* : European Journal of operationa research (2006)
- Glover,J.K.F and Laguna,M., *Genetic algorithms and tabu search: Hybrids for optimization* : Computer and operations Research (1995)
- Godfrey C. Onwubolu, and B.V. Babu, *New Optimization Techniques in Engineering* : Springer (2003)
- Michael Pinedo, *Scheduling, Theory, Algorithms, and systems, third edition*: Springer (2002)
- Pei Chann Chang, Shih-Hsin Chen, and Chin-Yuan Fan, *A hybrid electromagnetism-like algorithm for single machine scheduling problem* : Elsevier Ltd. (2007)
- Solis, F.J, and Wets,R.J-B, *Minimization by random search techniques*: matematics of operation research (1981)
- Wu, P., Yang, W. H., and wei,N.C, *An Electromagnetism Algorithm of neural network analysis-An application to textile retail operation*: Journal of the chinese institute of industrial engineers (2004)
- Wu, P., Yang, W. H., and Hung, Y.Y., *The study of electromagnetism-like mechanism based fuzzy neural network for learning fuzzi if-then rules*: Lecture notes in computer science (2005)
- X.Gandibleux, Marc S., Kenneth S, and Vincent T., *Metaheuristics for Multiobjective Optimisation*: Springer (2003)

APPENDIX 1

Example and Calculate Sum Earliness and Tardiness with EDD method, SPT Method and LPT method for jobs number are 5.

Job <i>i</i>	1	2	3	4	5
<i>d_i</i>	9	14	12	8	13
<i>p_i</i>	6	10	11	7	12

EDD (Earliest Due Date)

Job <i>i</i>	4	1	3	5	2	
<i>d_i</i>	8	9	12	13	14	
<i>p_i</i>	7	6	11	12	10	
<i>C_i</i>	7	13	24	36	46	
<i>E_i</i>	1	0	0	0	0	1
<i>T_i</i>	0	4	12	23	32	71
Z (1 E + 1 T)						72

SPT (Shortest Processing Time)

Job <i>i</i>	1	4	2	3	5	
<i>d_i</i>	9	8	14	12	13	
<i>p_i</i>	6	7	10	11	12	
<i>C_i</i>	6	13	23	34	46	
<i>E_i</i>	3	0	0	0	0	3
<i>T_i</i>	0	5	9	22	33	69
Z (1 E + 1 T)						72

LPT (Largest Processing Time)

Job <i>i</i>	5	3	2	4	1	
<i>d_i</i>	13	12	14	8	9	
<i>p_i</i>	12	11	10	7	6	
<i>C_i</i>	12	23	33	40	46	
<i>E_i</i>	1	0	0	0	0	1
<i>T_i</i>	0	11	19	32	37	99
Z (1 E + 1 T)						100

APPENDIX 2

Example and Calculate Sum Earliness and Tardiness with EDD method, SPT Method and LPT method for jobs number are 10.

Job <i>i</i>	1	2	3	4	5	6	7	8	9	10
<i>d_i</i>	10	15	14	8	16	9	5	12	11	7
<i>p_i</i>	6	13	14	7	12	8	4	10	9	5

EDD (Earliest Due Date)

Job <i>i</i>	7	10	4	6	1	9	8	3	2	5	
<i>d_i</i>	5	7	8	9	10	11	12	14	15	16	
<i>p_i</i>	4	5	7	8	6	9	10	14	13	12	
<i>C_i</i>	4	9	16	24	30	39	49	63	76	88	
<i>E_i</i>	1	0	0	0	0	0	0	0	0	0	1
<i>T_i</i>	0	2	8	15	20	28	37	49	61	72	292
Z (1 E + 1 T)											293

SPT (Shortest Processing Time)

Job <i>i</i>	7	10	1	4	6	9	8	5	2	3	
<i>d_i</i>	5	7	10	8	9	11	12	16	15	14	
<i>p_i</i>	4	5	6	7	8	9	10	12	13	14	
<i>C_i</i>	4	9	15	22	30	39	49	61	74	88	
<i>E_i</i>	1	0	0	0	0	0	0	0	0	0	1
<i>T_i</i>	0	2	5	14	21	28	37	45	59	74	285
Z (1 E + 1 T)											286

LPT (Largest Processing Time)

Job <i>i</i>	3	2	5	8	9	6	4	1	10	7	
<i>d_i</i>	14	15	16	12	11	9	8	10	7	5	
<i>p_i</i>	14	13	12	10	9	8	7	6	5	4	
<i>C_i</i>	14	27	39	49	58	66	73	79	84	88	
<i>E_i</i>	0	0	0	0	0	0	0	0	0	0	0
<i>T_i</i>	0	12	23	37	47	57	65	69	77	83	470
Z (1 E + 1 T)											470

APPENDIX 3

Example and Calculate Sum Earliness and Tardiness with EDD method, SPT Method and LPT method for jobs number are 15.

Job <i>i</i>	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
<i>d_i</i>	12	11	17	8	16	6	16	12	11	13	18	15	17	14	7
<i>p_i</i>	3	5	14	7	12	4	15	10	10	11	13	9	4	6	6

EDD (Earliest Due Date)

<i>Job i</i>	6	15	4	2	9	1	8	10	14	12	5	7	3	13	11	
<i>d_i</i>	6	7	8	11	11	12	12	13	14	15	16	16	17	17	18	
<i>p_i</i>	4	6	7	5	10	3	10	11	6	9	12	15	14	4	13	
<i>C_i</i>	4	10	17	22	32	35	45	56	62	71	83	98	112	116	129	
<i>E_i</i>	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2
<i>T_i</i>	0	3	9	11	21	23	33	43	48	56	67	82	95	99	111	701
Z (1 E + 1 T)																703

SPT (Shortest Processing Time)

<i>Job i</i>	1	6	13	2	14	15	4	12	8	9	10	5	11	3	7	
<i>d_i</i>	12	6	17	11	14	7	8	15	12	11	13	16	18	17	16	
<i>p_i</i>	3	4	4	5	6	6	7	9	10	10	11	12	13	14	15	
<i>C_i</i>	3	7	11	16	22	28	35	44	54	64	75	87	100	114	129	
<i>E_i</i>	9	0	6	0	0	0	0	0	0	0	0	0	0	0	0	15
<i>T_i</i>	0	1	0	5	8	21	27	29	42	53	62	71	82	97	113	611
Z (1 E + 1 T)																626

LPT (Largest Processing Time)

<i>Job i</i>	7	3	11	5	10	8	9	12	4	14	15	2	6	13	1	
<i>d_i</i>	16	17	18	16	13	12	11	15	8	14	7	11	6	17	12	
<i>p_i</i>	15	14	13	12	11	10	10	9	7	6	6	5	4	4	3	
<i>C_i</i>	15	29	42	54	65	75	85	94	101	107	113	118	122	126	129	
<i>E_i</i>	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
<i>T_i</i>	0	12	24	38	52	63	74	79	93	93	106	107	116	109	117	1083
Z (1 E + 1 T)																1084

APPENDIX 4

Example and Calculate Sum Earliness and Tardiness with EDD method, SPT Method and LPT method for jobs number are 20.

Job <i>i</i>	1	2	3	4	5	6	7	8	9	10
<i>d_i</i>	16	11	17	8	13	6	20	12	11	13
<i>p_i</i>	3	6	5	7	12	4	18	10	9	11

Job <i>i</i>	11	12	13	14	15	16	17	18	19	20
<i>d_i</i>	18	15	17	14	7	5	19	22	18	21
<i>p_i</i>	16	9	4	13	6	2	15	17	13	19

EDD (Earliest Due Date)

Job <i>i</i>	16	6	15	4	2	9	8	5	10	14
<i>d_i</i>	5	6	7	8	11	11	12	13	13	14
<i>p_i</i>	2	4	6	7	6	9	10	12	11	13
<i>C_i</i>	2	6	12	19	25	34	44	56	67	80
<i>E_i</i>	3	0	0	0	0	0	0	0	0	0
<i>T_i</i>	0	0	5	11	14	23	32	43	54	66
Z (1 E + 1 T)										

12	1	3	13	11	19	17	7	20	18	
15	16	17	17	18	18	19	20	21	22	
9	3	5	4	16	13	15	18	19	17	
89	92	97	101	117	130	145	163	182	199	
0	0	0	0	0	0	0	0	0	0	3
74	76	80	84	99	112	126	143	161	177	1380
										1383

SPT (Shortest Processing Time)

Job <i>i</i>	16	1	6	13	3	2	15	4	9	12
<i>d_i</i>	5	16	6	17	17	11	7	8	11	15
<i>p_i</i>	2	3	4	4	5	6	6	7	9	9
<i>C_i</i>	2	5	9	13	18	24	30	37	46	55
<i>E_i</i>	3	11	0	4	0	0	0	0	0	0
<i>T_i</i>	0	0	3	0	1	13	23	29	35	40
Z (1 E + 1 T)										

8	10	5	14	19	17	11	18	7	20	
12	13	13	14	18	19	18	22	20	21	
10	11	12	13	13	15	16	17	18	19	
65	76	88	101	114	129	145	162	180	199	
0	0	0	0	0	0	0	0	0	0	18
53	63	75	87	96	110	127	140	160	178	1233
										1251

LPT (Largest Processing Time)

Job i	20	7	18	11	17	14	19	5	10	8
d_i	21	20	22	18	19	14	18	13	13	12
p_i	19	18	17	16	15	13	13	12	11	10
C_i	19	37	54	70	85	98	111	123	134	144
E_i	2	0	0	0	0	0	0	0	0	0
T_i	0	17	32	52	66	84	93	110	121	132
$Z(1E + 1T)$										

9	12	4	2	15	3	6	13	1	16	
11	15	8	11	7	17	6	17	16	5	
9	9	7	6	6	5	4	4	3	2	
153	162	169	175	181	186	190	194	197	199	
0	0	0	0	0	0	0	0	0	0	2
142	147	161	164	174	169	184	177	181	194	2400
										2402

