OPTIMIZATION OF TIME-COST-RESOURCE TRADE-OFF PROBLEMS IN PROJECT SCHEDULING USING META-HEURISTIC ALGORITHMS

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY
ÖNDER HALİS BETTEMİR

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF DOCTORATE OF PHILOSOPHY
IN
CIVIL ENGINEERING

AUGUST 2009

Approval of the thesis:

OPTIMIZATION OF TIME-COST-RESOURCE TRADE-OFF PROBLEMS IN PROJECT SCHEDULING USING META-HEURISTIC ALGORTITHMS

submitted by **ÖNDER HALİS BETTEMİR** in partial fulfillment the requirements for the degree of **Doctorate of Philosophy in Civil Engineering Department, Middle East Technical University** by,

Prof. Dr. Canan Özgen ____________
Dean, Graduate School of **Natural and Applied Sciences**

Prof. Dr. Güney Özcebe ____________
Head of Department, **Civil Engineering**

Assoc. Prof. Rıfat Sönmez ____________
Supervisor, **Civil Engineering Dept., METU**

Examining Committee Members:

Prof. Dr. M. Talat Birgönül ____________
Civil Engineering Dept., METU

Assoc. Prof. Rıfat Sönmez ____________
Civil Engineering Dept., METU

Assist. Prof. Metin Arıkan ____________
Civil Engineering Dept., METU

Prof. Dr. Cevriye Gencer ____________
Indistrial Engineering Dept., Gazi University

Assoc. Prof. Murat Gündüz ____________
Civil Engineering Dept., METU

Date: 13.08.2009

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last name: Önder Halis BETTEMİR

Signature :

ABSTRACT

OPTIMIZATION OF TIME-COST-RESOURCE TRADE-OFF PROBLEMS IN PROJECT SCHEDULING USING META-HEURISTIC ALGORITHMS

Bettemir, Önder Halis
PhD., Department of Civil Engineering
Supervisor : Assoc. Prof. Rıfat Sönmez

August 2009, 227 pages

In this thesis, meta-heuristic algorithms are developed to obtain optimum or near optimum solutions for the time-cost-resource trade-off and resource leveling problems in project scheduling. Time cost trade-off, resource leveling, single-mode resource constrained project scheduling, multi-mode resource constrained project scheduling and resource constrained time cost trade-off problems are analyzed.

Genetic algorithm simulated annealing, quantum simulated annealing, memetic algorithm, variable neighborhood search, particle swarm optimization, ant colony optimization and electromagnetic scatter search meta-heuristic algorithms are implemented for time cost trade-off problems with unlimited resources. In this thesis, three new meta-heuristic algorithms are developed by embedding meta-heuristic algorithms in each other. Hybrid genetic algorithm with simulated annealing presents the best results for time cost trade-off.

Resource leveling problem is analyzed by five genetic algorithm based meta-heuristic algorithms. Apart from simple genetic algorithm, four meta-heuristic algorithms obtained same schedules obtained in the literature. In addition to this, in one of the test problems the solution is improved by the four meta-heuristic algorithms.

For the resource constrained scheduling problems; genetic algorithm, genetic algorithm with simulated annealing, hybrid genetic algorithm with simulated annealing and particle swarm optimization meta-heuristic algorithms are implemented. The algorithms are tested by using the project sets of Kolisch and Sprecher (1996). Genetic algorithm with simulated annealing and hybrid genetic algorithm simulated annealing algorithm obtained very successful results when compared with the previous state of the art algorithms.

120-activity multi-mode problem set is produced by using the single mode problem set of Kolisch and Sprecher (1996) for the analysis of resource constrained time cost trade-off problem. Genetic algorithm with simulated annealing presented the least total project cost.

Keywords: Meta-heuristic algorithm, Planning, Optimization, Time cost trade-off, Resource Leveling, Resource constraint project scheduling, Resource constraint time cost trade-off problem

ÖZ

MODERN-SEZGİSEL YÖNTEMLERLE PROJE PLANLAMASINDA ZAMAN-MALİYET-KAYNAK ÖDÜNLEŞİM PROBLEMLERİNİN OPTİMİZASYONU

Bettemir, Önder Halis

Doktora, İnşaat Mühendisliği Bölümü

Tez Yöneticisi : Doç. Dr. Rıfat Sönmez

Ağustos 2009, 227 sayfa

Bu tez çalışmasında, proje planlaması ile ilgili zaman-maliyet- kaynak ödünleşim ve kaynak dengeleme problemlerinin en iyi veya yakın en iyi sonuçlarının bulunabilmesi için modern sezgisel yöntemler geliştirilmiştir. Bu amaçla, kaynak dengelemesi, sınırlı ve sınırsız kaynaklı projelerin zaman maliyet analizi, sınırlı kaynaklı tek yapım ve çok yapım yöntemli projelerin zaman çizelgelerinin hazırlanması problemleri incelenmiştir.

Literatürdeki Genetik algoritma (GA), tavlama benzetimi, kuantum tavlama benzetimi, deneyimsel algoritma, komşu arama, kuş sürüsü optimizasyonu, karınca koloni optimizasyonu ve elektromanyetik saçılım algoritmaları zaman maliyet problemi çözümü için uygulanmıştır. Bu modern sezgisel algoritmalardan üç yeni melez modern sezgisel yöntem geliştirilmiştir. Sabit maliyetli zaman maliyet analizi için, melez genetik algoritma tavlama benzetimi yöntemi en iyi sonucu vermiştir.

Kaynak dengeleme problemi için, GA ve genetik algoritma tabanlı modern sezgisel yöntemler incelenmiştir. GA dışındaki dört yöntemle, literatürdeki sonuçlarla aynı sonuçlar elde edilirken; test problemlerinin birinde mevcut çözümlerden daha iyi sonuç elde edilmiştir.

Sınırlı kaynaklı proje planlaması problemleri için; GA, genetik algoritma tavlama benzetimi, melez genetik algoritma tavlama benzetimi ve kuş sürüsü algoritması literatürdeki test örnekleri ile denenmiştir. Sonuçlar karşılaştırıldığında, Genetik algoritma tavlama benzetimi ve melez genetik algoritma tavlama benzetimi algoritmalarının daha iyi çözümler verdiği görülmüştür.

Sınırlı kaynaklı zaman maliyet analizinde kullanılmak üzere literatürdeki 120 aktiviteli tek yapım yöntemli proje setinden, çok yapım yöntemli problem seti elde edilmiştir. Bu analizde genetik algoritma tavlama benzetimi yöntemi en iyi sonucu vermiştir.

Anahtar Kelimeler: modern sezgisel algoritmalar, en iyi sonuç, planlama, zaman maliyet analizi, kaynak dengeleme, kısıtlı kaynaklı proje planlaması, kısıtlı kaynaklı zaman maliyet analizi

ACKNOWLEDGEMENTS

The author wishes to express his deepest gratitude to his supervisor Assoc. Prof. Rıfat Sönmez for his guidance, advice, criticism and insight throughout the research.

The author owes special thanks to Prof. Dr. Cevriye Gencer for her special comments and suggestions.

Special thanks to Prof. Dr. M. Talat Birgönül, Assist. Prof. Metin Arıkan and Assoc. Prof. Murat Gündüz for their advices, comments and believing in my determination on the subject and their patience.

The author would also like to thank Kerem Tanboğa for his suggestions, encouragement and comments.

Özlem Yavuzyılmaz shared her knowledge on word editing and she had spent most of her time for doing thesis formatting, thanks for her help. It was really impossible for me to do most of the thesis formatting alone.

Wherever in the world you have been, first you watch Lig TV and than you thank to Emre Caner Akçay. Thanks for Onur Çelenligil, Armin and Matineh for their moral support and encouragements.

Many thanks to Gülşah Fidan for printing the thesis, she helped me enormously.

Finally endless thanks to my father Veli Bettemir, my mother Nazife Bettemir and my Sister Naciye Neşe Bettemir for their support and encouragements during the thesis study.

To My Father,

Veli BETTEMİR,

TABLE OF CONTENTS

LIST OF TABLES

TABLES

LIST OF FIGURES

FIGURES

LIST OF ABBREVIATIONS

ACO	Ant Colony Optimization
AIS	Artificial Immune System
AoN	Activity on Node
B&B	Branch and Bound
BC	Boltzmann Constant
B&C	Branch and Cut
CB	Current Best
CH	Chromosome
CPM	Critical Path Method
CPU	Central Processing Unit
DE	Differential Evaluation
DF	Delayed Finish
DL	Decision Limit
DP	Dynamic Programming
DS	Delayed Start
DYNASTRAT	Dynamic strategy model
EF	Early Finish
EM	Electromagnetism Mechanism
ES	Early Start
ESS	Electromagnetic Scatter Search
FF	Finish to Finish
F&F	Filter and Fan
FS	Finish to Start
GA	Genetic Algorithm
GASA	Genetic Algorithm with Simulated Annealing
GASAVNS	Genetic Algorithm with Simulated Annealing and Variable Neighborhood Search
GHz	Giga Hertz

GMASA	Genetic Memetic Algorithm with Simulated Annealing
GRD	Greatest Resource Demand
GRU	Greatest Resource Utilization
HGAQSA	Hybrid Genetic Algorithm with Quantum Simulated Annealing
HGASA	Hybrid Genetic Algorithm with Simulated Annealing
IP	Integer Programming
LB	Lower Bound
LBP	Precedence-based Lower Bound
LBR	Resource-based Lower Bound
LFT	Minimum Late Finish Time
LF	Late Finish
LP	Linear Programming
LP/IP	Linear Programming Integer Programming
LS	Late Start
MA	Memetic Algorithm
MB	Megabyte
MINSLK	Minimum Activity Slack
MJP	Most Jobs Possible
MOGA	Multi Objective Genetic Algorithm
MRCPSP	Multi-mode Resource Constraint Project Scheduling Problem
MRPL	Maximum Remaining Path Length
PACK	Packing Method for Resource Leveling
PSO	Particle Swarm Optimization
PSPLIB	Project Scheduling Problem Library
QSA	Quantum Simulated Annealing
RAM	Random Access Memory
RAN	Random Activity Selection
RCPSP	Resource Constraint Project Scheduling Problem
RSM	Resource Selection Method
RWNo	Random Walk Number
SA	Simulated Annealing
SAperiod	Simulated Annealing period
SF	Start to Finish

SIO	Shortest Imminent Operations
SRCPSP	Single-mode Resource Constrained Project Scheduling Problem
SS	Start to Start
TCT	Time Cost Trade-off
UB	Upper Bound
VNS	Variable Neighborhood Search

CHAPTER 1

INTRODUCTION

Competition in the construction industry is increasing day by day as new firms are entering into market and the existing companies are enlarging their job opportunities by entering into new construction sectors. To gain competitive advantage against rivals, the construction companies aim to minimize the resource costs by means of minimizing the idle machinery and labor time which requires excellent planning and scheduling of construction projects.

Project planning, resource constrained scheduling and resource leveling has significant importance, since these tasks directly effect project completion duration and cost. To gain competitive advantage in the market, the project must be executed in the optimum or near-optimum planned state. In other words, to be strong against rivals optimum or near optimum solutions of time cost trade-off, resource leveling and resource constrained scheduling problems should be obtained in the planning phase of the project.

In this thesis, optimum or near optimum solution for time cost trade-off (TCT) problem with unlimited resources, resource leveling, resource constrained scheduling and time cost trade-off problem with limited resources are searched by meta-heuristic algorithms.

1.1 Background of this Research

This research aims to develop an optimization tool for the mentioned time-cost-resource based project planning and optimization problems. The problems in concern have a common point in which all have the objective function as minimization of an objective value. Total project cost for the TCT problem, fluctuations of resource demand for resource leveling and project duration for resource constrained scheduling are the objectives to be minimized.

Project duration can often be shortened by accelerating some of its activities at an additional expense. Crashing of activities increases the activity's cost, thus the direct project cost. However, crashing of activities reduces the project duration and decreases the indirect project costs. Summation of direct and indirect project costs are aimed to be minimized which is called as TCT problem (Hegazy 1999).

Over-timing or assigning more crew and equipment decreases the productivity. Thus unit work done per machine or labor decreases and total labor and equipment costs increase. Resource availabilities are not considered in the solution procedure of the TCT problem with unlimited resources. Optimum solution of TCT with unlimited resources minimizes the total project cost.

Fluctuations in the resource usage decrease the productivity because fluctuations causes idle labor and machinery during low resource demanding periods of the project. To prevent idle labor if only firing labors are preferred than during the execution of the project too much labor hiring and labor firing will occur. This may cause additional problems such as excessive decrease in production, problem in hiring labor during certain periods. Similarly, if idle equipment and machinery is aimed to be prevented by returning the rented machinery and renting it again when needed may cause additional problems. Increased transportation costs, problems in delivering the machine on time and lack of available machinery to rent in certain periods are some of the problems to be faced for frequent renting and returning equipment. As a result, resource demand profile is aimed to be smoothed as much as possible in order to minimize idle time of the resources. This problem type is called *resource leveling*. Resource leveling does not try to minimize the peak resource demand on purpose and assumes no resource limitation on the availabilities of the resources. However, at the end of the resource leveling, decrease in the maximum demand might be achieved.

There can be limitations on the hiring of certain labor types or renting of equipments. Besides, there can be limitations on accommodation of the labors in the construction site. As a result of these, there can be limitations on the maximum number of employed labor and/or on the hired machinery. In most of the cases, resource demand profile of the project obtained by taking early start times of the activities into account overrides the resource limits. Consequently, delays in some of the activities are unavoidable, if these activities are on the critical path, than project duration increases compared with the case of unlimited resources. Aim of resource constrained project scheduling problem is to complete the project in minimum duration without overriding the resource limitations. Optimum solution of resource constrained project scheduling problem gives the shortest project completion duration which satisfies the resource constraints and CPM relationships.

In this study, TCT, resource leveling and resource constrained scheduling problems will be handled. In this respect, optimum or near-optimum solutions of these problems are aimed. In order to achieve this task, meta-heuristic optimization algorithms are implemented.

Formulation of resource leveling or resource allocation is significantly difficult by means of polynomial functions or linear equations. Similarly, heuristic algorithms are problem dependent and can easily get stuck into local minima. Consequently, meta-heuristic algorithms are preferred for the solution method of the optimization problems.

1.2 Prospects from this thesis

Obtaining optimum or near-optimum solution for the project planning problems consisting of simple TCT, resource leveling and resource allocation is the main objective of this research. Meta-heuristic algorithms are implemented for the search of optimum or near optimum solution. Optimization procedure is performed by software generated during the thesis study. The software is developed in a way capable of performing CPM scheduling for the four kinds of logical relationships; FS, FF, SS, SF. Furthermore, positive or negative lags can be assigned to the relationships between the activities. The computer software is written and compiled by Microsoft .NET Visual Studio C++ and Microsoft Visual Studio 2008 C# programming software.

Genetic Algorithm (GA), Genetic Algorithm with Simulated Annealing(GASA), Hybrid Genetic Algorithm with Simulated Annealing (HGASA), Hybrid Genetic Algorithm with Quantum Simulated Annealing (HGAQSA), Genetic Memetic Algorithm with Simulated Annealing (GMASA), Genetic Algorithm with Simulated Annealing and Variable Neighborhood Search (GASAVNS), Particle Swarm Optimization (PSO), Ant Colony Optimization (ACO) and Electromagnetic Scatter Search (ESS) meta-heuristic algorithms are implemented and analyzed for the solution of optimization problem.

This thesis consists of solution of different type of planning problems. In the first task, TCT type problems with unlimited resources are analyzed. For the analysis, three case problems obtained from the literature are used. In addition to this, a 63-activity network is generated as a test problem.

Second task of the thesis study is the resource leveling. Similar to the previous task, sample problems obtained from literature is used and the results are compared by the results obtained from the literature. In addition to this, convergence speed of the algorithms is also taken into account.

Third task involves solution of resource constrained scheduling problem. In this task, shortest construction duration is searched by taking limited resources into account with one execution mode for each activity. Randomly generated networks obtained from PSPLIB are used as test problems.

Fourth task consists of resource constrained project scheduling with multi-mode activity execution modes. Solution algorithm aims to minimize project duration by choosing the optimum activity priorities and their execution modes. Performances of the algorithms are tested by JXX and RX multi-mode resource constrained problem sets obtained from PSPLIB.

Fifth task consists of optimization of project cost with limited resources for the projects with multi activity execution modes. The solution aims to minimize the summation of the direct and indirect costs without overriding the resource restrictions.

1.3 Scope and Limitations

In this thesis, it is aimed to obtain global optima for the solution of planning problems. However, if the number of activities of the project increases, the search space enlarges significantly which requires excessive number of iterations. Thus, computation time of the solution for global optima would be longer than reasonable duration. For this reason, near optimum solutions are aimed to be obtained for large projects in order to limit computation duration.

1.4 Organization of Thesis

In the second chapter of this thesis; CPM, project progress monitoring and tender types will be briefly explained. In the third chapter, meta-heuristic algorithms implemented in this thesis study are explained. Their implementation for the solution of optimization problem is illustrated. In chapter four, TCT analyzes are performed. Sample problems and analysis results are illustrated. Chapter five consists of the analysis related with resource leveling problems. In chapter six, resource constrained project scheduling with single activity execution mode is included. Chapter seven includes the resource constrained project scheduling problem with multi mode activity execution modes. TCT problem with limited resources is analyzed in chapter eight and an overall conclusion of the thesis is performed in chapter nine.

CHAPTER 2

PROJECT PLANNING, TIME COST TRADE-OFF AND RESOURCE ALLOCATION PROBLEMS

In this chapter, the basics of project scheduling, time cost trade-off (TCT) and resource leveling and resource constraint project scheduling are explained. Well known project scheduling method CPM and the topics related with the CPM is briefly explained. Objectives of the resource leveling and resource allocation problems and the related studies are mentioned.

2.1 CPM

Critical Path Method (CPM) was invented for planning and scheduling of projects. The planning and scheduling of construction activities has vital importance because the amount and the time of the resource and material requirements will be known before the commencement of the project. The benefits of using CPM can be briefly explained as (Suhanic 2001):

- CPM pinpoints the activities whose completion times are responsible for establishing the overall project duration. Identification of critical activities helps to pay more attention on these activities to keep them on schedule.
- CPM gives a quantitative evaluation of the amount of float that each activity has. Within the limits of float time the activities with float may be started and finished later than the earliest dates, or they may be shifted in time to smooth labor or equipment requirements. This property of CPM gives the data for resource leveling.
- CPM shows the most economical scheduling for all activities for each possible project completion date. This allows consideration of both time and cost in choosing methods, equipment, materials, crews, and work hours.

- CPM provides the necessary data for choosing the best project completion date which is used for TCT analyses.
- Effect of changing the activity execution modes is monitored effectively by CPM networks.

CPM can be implemented on two different network types: activity on arrow and activity on node. As their name implies, in activity on arrow diagram the activities are represented by the arrows and the arrows are connected to the nodes which are events. In activity on node, the activities are represented by nodes and the logical relationships between the activities are demonstrated by the arrows.

In this thesis, the scheduling of the construction activities is performed by activity on node diagrams. The reason of this selection is, construction of AoA diagrams requires considerably more endeavor than construction of AoN diagrams and there is less data requirement from the user. In addition to this, activity on node diagrams are suitable for defining logical relationships and lags. Terminology for the CPM planning is represented below (Suhanic 2001):

Activity: Discretely defined task. Activities are specified by an activity number, description, duration, and type. Furthermore, cost and resource demand may also be represented.

Critical Path: Longest path through a schedule network, or the chain or sequence of activities that takes the longest time, or the longest irreducible sequence of events. The critical path determines the project duration.

Duration: Time period expressed in working days between the start and finish of an activity.

Early Start Date (ES): The earliest date that and activity can start based on the logical relationships among its predecessors.

Early Finish Date (EF): The earliest date that an activity can finish based on its duration, and logical relationships among its predecessors.

Late Start Date (LS): Latest start date calculated for an activity on the backward pass. Late start is the activity latest start date allowed so as not to delay the project completion date.

Late Finish (LF): Latest finish date calculated for an activity on the backward pass. Late finish is the activity latest finish date allowed so as not to delay the project completion date.

Finish to Finish (FF): Network relationship whereby the finish of a preceding activity is a condition for the finish of the succeeding activity.

Finish to Start (FS): Network relationship whereby the preceding activity must finish before the succeeding activity can start.

Start to Finish (SF): Network relationship whereby the activity can only finish if its predecessor activity has started.

Start to Start (SS): Network relationship whereby the activity can only start if its predecessor activity has started.

Total Float: Amount of leeway that an activity has in the schedule before it adversely affects the critical path.

Free Float: The amount of leeway an activity has before it adversely affects another activity.

Forward Pass: Early start and early finish dates of all activities are calculated. The longest sequence of activities sets the critical path and the project completion date.

Backward Pass: Late start and late finish dates for all activities are determined by calculating backwards from the project end date, set by the forward pass calculation, to its beginning. Activities that have the same early and late start or same early and late finish dates are on the critical path. These activities are called critical activity.

2.2 TCT

Time Cost Trade off analysis is the compression of the project schedule to achieve a more favorable outcome in terms of project duration, cost, and projected revenues. The objectives of the TCT analysis are to compress the project to the optimum duration which minimizes the total project cost.

TCT analysis is performed by evaluating the possible crashing alternatives of the activities. Possible crashing durations can be obtained in three ways: by over sizing the crew, by over timing or by executing another construction technique which is faster but more expensive.

Total project cost consists of two parts, direct costs and indirect costs. Direct costs are the activity based costs which are the labor, material, equipment and machinery costs of the project. As the activity's normal duration is crashed, it is expected that the direct costs are increased. This is because in order to finish a certain activity, larger crews are assigned or the crew is over-timed. Larger crew size or over-timing decreases the productivity. As a result of this, unit cost per unit output increases. Indirect costs are the overhead costs and possible delay penalties of the projects. Overhead cost includes any costs of the project which can not be associated with the activities of the project. Salaries of the cook, security staff and office staff can be counted as overhead costs. Heating or cooling, illuminating of the site and the barracks are also included in overhead costs of the project. Overhead costs fluctuate during the project as it affected by the climate and the number of the workers in the site. In order to simplify the cost computations, overhead cost is usually assumed constant during the project. Consequently, indirect cost decreases when the project duration decreases.

TCT is one of the major interests of the construction management, since the optimum solution of TCT problems directly increases the productivity thus the profit of the project. As this is the case, several algorithms and heuristics are developed and implemented which aims to achieve the optimum solution of TCT problems. Importance of the TCT problem was recognized for approximately half a century ago, almost simultaneously with the development of project analysis techniques by Fulkerson and Kelly (De et al. 1995). First considerable attempt to solve TCT problem can be counted as the heuristic algorithm derived by Nicolai Siemens (Siemens 1971) and later his algorithm is improved by Goyal (1975, 1996).

This heuristic algorithm was based on crashing of the longest path(s) of the network if total crashing cost was less than the savings obtained by the reduction of indirect costs. The drawback of this algorithm is the requirement of the determination of all paths through the network. This requirement can easily be satisfied for a small sized project but for large projects, it would be impossible to store all paths in the memory of the computer. The number of path in the network grows exponentially as some of the activities have more than one predecessor. If some of the paths are eliminated by means of a heuristic algorithm, it can be possible to be stuck in local minimum, as a result heuristic methods are infeasible to implement if global minima is intended. Furthermore, this heuristic method is suitable for continuous crashing functions. If discrete crashing functions are assigned to the activities, the heuristic algorithm would prefer the least crashing cost sloped activity. If the activity have to be crashed more than the difference between the critical path and the second longest path, than the former critical path would be crashed unnecessarily longer. Other activities on the previous critical path might provide cheaper crashing alternatives which could end up with the same project length. To sum up, heuristic methods can be considered as a satisfactory tool for solution of TCT problems as they can provide near optimum solutions with reasonable computational duration. Consequently, many researchers implement heuristic algorithms in their studies for the search of the optimum solution of TCT problem (Panagiotafopoulos 1977, Schwarze 1980, Barber and Boardman 1988, Chiu and Chiu 2005, Vanhoucke and Debels 2007).

Some heuristic methods were also generated to solve time cost trade-off problems. These methods provide good solutions, but do not guarantee optimality. Fondahl's method (1961), Siemens's model (1971), and Moselhi's model (1993) are examples of heuristic approaches.

Many researchers attempted to solve TCT by linear programming (Babu and Suresh 1996, Burns et al. 1996, Khang and Myint 1999, Wei and Wang 2003, Moussourakis and Haksever 2004, Vanhoucke 2005, Yang 2005a, 2005b, Bidhandi 2006). The advantage of Linear Programming (LP) can be seen as any kind of TCT problem can be converted into LP and solved by using commercial linear programming software. If the cost function includes quadratic equations, the quadratic equations are linearized by Taylor series expansion and the LP is solved iteratively. The disadvantage of the LP is that the number of parameters grows considerably as the number of activities increases. Parameter number to be solved is around four to five times the number of activities depending on the TCT problem which makes the solution of TCT problem difficult for large sized projects. If there is not any memory limitation on the computer, it is guaranteed to obtain optimum solution of TCT by LP.

The nature of the TCT problems is suitable for the solution of linear programming. It is expected that the cost slope of activities increase as the activities are further crashed. This is mainly caused by the reduced productivity of labor and machinery as the crew size or amount of overtime is increased. As a result of this each activity would have an increasing sloped crash duration versus crash cost curve. As a result of this, the realistic crashing alternatives end up with convex solution space which guarantees the solution algorithm of LP to converge into global optimum.

Dynamic Programming (DP) is a talented method which reduces the network size significantly by merging the activities. DP aims to reduce the network to one node system and solves the TCT problem according to the one node system's crashing alternatives. The reduction algorithm can be programmed but keeping or eliminating the crashing alternatives of the reduced alternatives is difficult and requires high storage capacity as activities are reduced for large projects. In addition to this, every network can not be reduced to a one node system because of complex logical relationships between the activities. Merging of the alternatives are not possible if there are complex relationships and those nodes are kept as they are. At the end of the merging process, the network is solved by exhaustive enumeration of the complex nodes which can not be merged. Computation duration of DP is reasonable for small and simple networks, however for the complex networks exhaustive enumeration may not be practical. De et al. 1995 and Demeulemeester et al. 1996 clearly illustrated and explained the decomposition algorithms of the network.

Mathematical programming methods were generally used to solve time cost trade-off problems. The methods used either linear programming or dynamic programming (Kelly 1961, Meyer and Shaffer 1965, Butcher 1967, Talbot 1982). In these methods, the relationships between activity costs and durations are generally assumed as: (1) linear or nonlinear; (2) concave, convex, or not fixed; (3) discrete or continuous; or (4) hybrid.

Ant Colony Optimization (ACO) had been implemented by Kuang and Xiong (2005). They solved a small project with 7 activities and find the global optimum with ACO by searching only 6.63% of the possible search space. Afshar et al.

(2007), Ng and Zhang (2008) and Xiong and Kuang (2008) also analyzed TCT problem by ACO meta-heuristic algorithms. Ng and Zhang (2008) embedded a local search algorithm to the ACO.

Afshar et al. (2009) proposed Nondominated Archiving ACO (NA-ACO) algorithm in which all ant colonies are initiated by the same number of ants and arbitrary order is given to the colonies. Ants in a certain colony simultaneously explore a solution according to the objective assigned to that colony. Solutions found for one objective in one cycle are evaluated in the next colony according to the competing objective assigned to that colony. If there is an improvement the optimal path is updated. Afshar et al. (2009) used 18-activity project analyzed by Hegazy (1999) before. The project is analyzed for several indirect project costs.

Genetics Algorithm (GA) is applied for the solution of TCT by many researchers since GA is a good candidate for finding the global optimum (Zheng et al. 2004, Li and Love 1997, Li et al. 1999, Zheng et al. 2005, Feng et al. 1997, Eshtehardian et al. 2008, Elbeltagi et al. 2005). The advantages of GA can be counted as its ability to manage any kind of crashing function such as discrete, linear, and nonlinear, can easily be programmed and can systematically surfs through the search space to avoid local minimum. However, as the number of activities increases the number of evaluation should be increased in order to obtain better results increasing the computation time increases as well.

Meta-heuristic algorithms are widely applied for the solution of simple TCT problem, although LP is guaranteed to solve simple TCT problem. One reason for the application of meta-heuristic algorithm is that, the LP is a difficult method to apply for a planning engineer. The linear equations and definition of slack variables require skilled planners. However, there is not any requirement of a-priori information if meta-heuristic algorithms are implemented. Only definition of crashing alternatives and CPM network is enough.

Meta-heuristic algorithms are easy to implement, but they can not always guarantee optimum solution. The largest project analyzed by meta-heuristic algorithm found in the literature was 18-activity project. Even the analyzes of 18-activity project did not obtain the optimum solution at each trial (Elbeltagi et al. 2005). In this situation the meta-heuristic algorithms becomes undependable for the optimization of TCT problems.

Main deficiency of the existing meta-heuristic algorithms is that the algorithms could not improve the current best even the iteration number is increased. This is because the meta-heuristic algorithms get stuck into local optima and could not escape. The other reason is the imperfections in the search algorithms that the optimum search is mainly based on random changes committed on the individuals. The search algorithm should involve conscious optimum search process.

In order to improve the convergence capability of the algorithms, hybrid meta-heuristic algorithms are developed in this thesis study. Genetic algorithm, simulated annealing, memetic search algorithm and variable neighborhood search algorithms are the meta-heuristic methods developed for improving the convergence capability.

Optimum solution of simple TCT problem by meta-heuristic algorithms can be seen as an inessential endeavour, as the guaranteed optimal solution can be obtained by LP. However, the main reason of the analysis of simple TCT problems with meta-heuristic algorithms is to use the TCT problems as a tool which would help to monitor the progress obtained in the convergence capacity.

Instead of using sinusoidal functions which are theoretical testing examples, it is preferred to use realistic project planning example problems. TCT is a good candidate for this purpose as the optimum solution of the problem can be obtained by LP. In addition to this, the problem characteristics of TCT are similar to resource levelling and resource allocation problems. If a meta-heuristic algorithm successfully converges to the global optima in TCT problem, than it can be expected that the meta-heuristic algorithm will also converge to optimum or near-optimum in resource levelling and resource allocation problems. However, if a meta-heuristic algorithm converges into global optima in sinusoidal test functions, it can not be soundly expected that the algorithm will also converge to optimum in resource levelling and resource constrained scheduling problems.

In addition to this, knowledge of global optima of the test function has significant importance. Otherwise, only relative improvements would be able to be monitored by comparing the results of the other meta-heuristic methods. However, by this way absolute convergence capability of the meta-heuristic algorithms would be able to be monitored.

2.3 Resource Leveling

The resource leveling problem arises when there are sufficient resources available and it is necessary to reduce the fluctuations in the resource usage over the project duration. The objective of the leveling process is to “smooth” resource usage profile of the project without elongating the project duration as much as possible. This is accomplished by rescheduling of activities within their available slack to give the most acceptable profiles (Davis E. W. 1973). In resource leveling, the project duration of the original critical path remains unchanged (Senouci and Adeli 2001).

Fluctuations of resources are undesirable for the contractor for two reasons. It is expensive to hire and fire labor on a short term basis to satisfy fluctuating resource requirements. Resources can not be managed efficiently, if the schedule demands more output per day than possible with available resources (Son and Skibniewski 1999). Efficient use of project resources will decrease construction costs to owners and consumers, and at the same time, will increase contractor’s profits (Hegazy and Kassab 2003). In other words, alternative labor utilization strategies and better utilization of existing labor resources are needed to improve work productivity and reduce construction costs (Burleson 1997).

The peak demand and fluctuations of resources are undesirable for the contractor because: it is expensive to hire and fire labor on a short term basis to satisfy fluctuating resource requirements; resources cannot be managed efficiently if the schedule demands more output per day than possible with available resources (Harris 1978; Stevens 1990; Martinez and Ioannou 1993).

The availability of various resource utilization options at the activity level creates a very large number of possible combinations of resource utilization plans at the project level, where each is associated with a unique project duration and cost (Kandil and El Rayes, 2006).

Resource leveling method was introduced (Burgess and Killebrew 1962) in order to reduce the fluctuations in the resource profile. Noncritical activities are shifted within their available float to minimize the following objective function:

$$Z = \sum_{i=1}^{T} \left(y_i - \overline{y}_i\right)^2$$

Where T is the project duration; y_i is the sum of resource requirements of the activities performed at time unit i; and $\overline{y}_i$ is mean of the resource requirement during the project.

The Burgess procedure can be explained in following (Moder et al. 1983):

- List the activities in a certain priority, i.e. depending on total float, activity ID, free floats, activity duration or resource demand.
- Starting with the last activity listed according to the priority rule. Schedule the network by delaying the activity one period each time and compute the evaluation function. Select the schedule which gives the least evaluation function.
- Hold the previous activity fixed and repeat the previous step for the next activity. In order to search possible alternative the fixed activity is released if additional float is obtained for the present activity. If there is an improvement in the evaluation function, start and finish time of the activity is updated.
- The process is repeated until all activities are considered. This completes one cycle.

- Additional schedule is performed by starting from the last activity in the list which permits only delaying of the activities. If there is an improvement in the evaluation function, the whole cycle is repeated.

Application of Burgess procedure gives smoother resource profile but it is not guaranteed to obtain the possible least evaluation function.

In the past few decades, traditional resource optimization was based on either mathematical methods or heuristic techniques. Mathematical methods, such as integer, linear, or dynamic programming have been proposed for individual resource problems. Mathematical methods, however, are computationally non-tractable for any real life project, which is reasonable in size (Moselhi and Lorterapong 1993; Allam 1988). In addition, mathematical models suffer from being complex in their formulation and may be trapped in local optimum (Li and Love1997; Hegazy 2001). Heuristic methods, on the other hand, use experience and rules-of-thumb, rather than rigorous mathematical formulations. Despite their simplicity, heuristic methods perform with varying effectiveness when used on different project networks, and there are no hard guidelines that help in selecting the best heuristic approach to use. Therefore, the heuristic methods can not guarantee optimum solutions (Hegazy and Kassab 2003).

Hiyassat (2000) modified the minimum moment approach in resource leveling. The proposed method assumes limited project duration with an unlimited availability of resources. The final goal of the modification was to reduce the amount of calculations without sacrificing the accuracy of the results.

Son and Skibniewski (1999) introduced a multi-heuristic model called local optimizer and a hybrid model combining the local optimizer with simulated annealing for the solution of resource leveling problems.

Easa (1989) used integer programming techniques to solve the resource leveling problem. Integer linear programming procedures that have been developed for the resource leveling problem include (Ahuja, 1976; Easa, 1989; Elmaghraby, 1977;

Moodie and Mandeville, 1966). For example, Ahuja (1976) presents an integer linear programming formulation for minimizing resource variations between consecutive periods, and solves it using an explicit enumeration algorithm. Ahuja (1976) presented exhaustive enumeration procedures which permit the calculation of optimum solutions for resource leveling problems. However, the resource leveling problem has a phenomenon of “combinatorial explosion,” especially for large-scale problems. However, integer linear programming algorithm does not guarantee finding global optima of the evaluation function resource leveling. When the complexity of linear programming algorithm is taken into account, implementation of meta-heuristic algorithms for the solution of resource leveling problems will be clearly acceptable. To avoid the explosion problem, heuristic rules were mostly used to solve the problems (Easa 1989).

Major efforts for resource leveling heuristic procedures have been expended in developing rules which produce “good” feasible solutions. To date, many heuristic scheduling rules have been proposed to solve project scheduling problems. The PACK model (Harris R. B. 1990) and the NASTRAT model (Padilla and Carr 1991) are examples of heuristic methods. Wiest and Levy (1977), Antill and Woodhead (1982), Moder et el. (1983), have also developed heuristic rules for construction resource leveling problems. Mathematical models may guarantee optimal solutions on small-scale problems. However, it is difficult to create general mathematical models and extensive computational effort is required for larger problems (Leu S. S. et al. 2000).

Heuristic procedures developed for the resource leveling problem include those reported in Antill and Woodhead (1970), Burgess and Killebrew (1962), Harris (1978), Shaffer et al. (1965), Woodworth and Willie, (1975). The basic concept of these heuristics is to reschedule non-critical activities within the limits of available float according to some heuristic rule to achieve a better distribution of resource usage.

2.4 Single Mode Resource Constrained Project Scheduling

Resource-constrained scheduling arises when there are definite limits on the amount of resources available. The scheduling objective is to extend the project duration as little as possible beyond the original critical path duration in such a way that the resource constraints are met. In this process, both critical and noncritical activities are shifted.

Available resources are not unlimited and most of the time, initial schedule obtained from the CPM demands for excessive amounts of resources than available. In such cases, delaying the activities within the total floats may not help to decrease resource requirement to the available amounts. If this is the case, it will be impossible to complete the project within the latest finish time obtained by CPM.

If resources are not adequate for the commencement of an activity, the activity is delayed until the resources are adequate. The aim is to minimize the project elongation caused by the resource shortage. Priorities are assigned to the activities which demand for the same resources and best combination of activity priorities is searched via analytic or heuristic algorithms.

If there are limited amounts of resources available during each time period of project duration, the problem is called resource constrained project scheduling problem. When the amounts available are not sufficient to satisfy demands of concurrent activities, sequencing decisions are required, often with a resultant increase in project duration beyond the original Critical Path duration. While the most common objective is that of minimizing the increase in project duration, other objectives, such as cost minimization, are not unusual (Davis E. W. 1973).

If there are more than one alternative for activity duration and resource requirement, than the problem is called “multi mode resource constrained project scheduling”. In this problem both priority of activities and construction alternative of that activity are important.

There are two general methods for applying heuristics in project resource allocation problems. A serial scheduling procedure is one in which all activities of the project are ranked in order of priority as a single group, using some heuristic, and then scheduled. Activities that can not be started at their early start time are progressively delayed until sufficient resources are available.

In parallel scheduling, all activities starting in a given time period are ranked as a group in order of priority and resources allocated according to this priority as long as available. When an activity can not be scheduled in a given time period for lack of resources, it is delayed until the next time period. At each successive time period a new rank-ordering of all eligible activities is made and the process is continued until all activities have been scheduled (Moder et al. 1983).

Many researcher are focused on SRCPSP (Davis 1973, Woodworth and Willie 1975, Patterson 1984, Allam 1988, Al-jibouri 2002, Bock and Patterson 1990, Seibert and Gerald 1991, Dean et al. 1992, Chan et al. 1996, Nudtasomboon and Randhawa 1997, Savin et al. 1997, Savin et al. 1998, Mattila and Abraham 1998, Leu and Yang 1999, Brucker et al. 1999, Hiyassat 2000, Leu et al. 2000, Neumann and Zimmermann 2000, Senouci and Adeli 2001, Wei et al. 2002).

The zero-one integer linear programming model has been widely used to formulate the resource allocation problem (Elmaghraby and Cole 1963, Brand et al. 1964, Hadley 1964, Elmaghraby 1967, Pritsker et al.1969, Davis 1973, Wiest and Levy 1977). The earliest studies show that the LP is implemented for the solution of SRCPSP since 1960s. Early attempts to solve the resource-constrained scheduling problem concentrated in two areas: the formulation and solution of the problem as a mathematical programming problem, and the development of heuristic or approximate solution procedures for obtaining good or satisfying solutions to the problem. Numerous formulations of the problem were proposed but found to be impractical except for solving small problems of only a few activities. Wiest (1967) showed that in order to solve a 55-activity project with four resource types, solution of more than 6000 equations and 1600 variables is necessary.

Later researchers improved on the formulation techniques and solution of larger networks became possible. In addition to this, increase of the computer memory and CPU clock speed also helped application of LP for the solution of SRCPSP. The researchers improved the LP formulation (Patterson and Huber 1974, Lee et al. 1976, Hannan 1978, Slowinski 1981, Lee and Olson 1984, Mohanty and Siddiq 1989).

Patterson and Huber (1974) combined a minimum bounding procedure with LP to reduce the computation time required in arriving at minimum project duration. The algorithm starts the optimization procedure with a good lower bound solution to reduce the domain of possible solutions over which the LP algorithm must search.

To overcome the problems associated with optimization, special algorithms have been developed for solving the resource-constrained problems. These include the bounded enumeration approach (Davis and Heidorn, 1971); the branch and bound approach (Johnson, 1967; Stinson, 1976; Stinson et al., 1978); and the implicit enumeration approach (Patterson and Roth (1976), Talbot (1976), (1982), Talbot and Patterson, (1978)).

Enumeration techniques are based on enumeration of all possible activity sequencing combinations. The term Branch and Bound (B&B) refers to a generic type of optimization procedure which involves partitioning a problem into sub-problems. B&B can be modeled by the nodes and branches of a tree, to enumerate possible alternatives in arriving at the best solution. The results from these approaches vary depending on model complexity. However, techniques such as the Talbot's implicit enumeration algorithm (Talbot, 1976) have been shown to provide efficient solutions to the constrained project problem. Power of the algorithm comes from the facts that the enumeration process is implicit, in which some schedule alternatives can be immediately identified as not capable of leading to an improvement in the search for the optimum schedule. With this property, numerous schedule alternatives are eliminated without evaluating them.

Solution process of SRCPSP by B&B consists of branching, building up the tree; bounding, evaluating the nodes; pruning non-optimal portions of the tree. Nodes in the decision tree form unique partial schedules. Each partial schedule represents scheduling decisions for some subset of the total number of activities. The partial schedules are always feasible which satisfy both precedence and resource constraints. In addition to this, there is no redundancy in the schedule meaning that no two schedules are alike (Moder et al. 1983).

Tree generation process starts with creation of an initial node representing the set of activities which can be started at the beginning of the project. A family of partial schedules is created by branches with new nodes to the three starting from the first node. Each node in the family created from a particular node has in common with the others all scheduling decisions made previously in creating the common node. However each partial schedule is unique from the others is that it includes one new decision involving the scheduling of one or more activities previously unscheduled.

Therefore each branching operation creates as many new partial schedules as feasible combinations of activities that can enter the schedule in time, t_n. Thus a partial schedule, PS_n, can be visualized as a real project in progress at time t_n where some activities, the complete set C_n, will have been completed at t_n and the others, active set A_n, are actively in progress and have to be finished at a later date. To prevent growth in the number of branches, infeasible partial projects are pruned.

At any point in the project scheduling, a minimum length schedule for the remaining, unscheduled, activities can be calculated by ignoring possible resource conflicts. In this case, the remaining of the project is scheduled by only considering the activity durations and precedence requirements. The completion time of this path constitutes a lower bound on completion time of any partial schedule derived from this partial schedule. Lower bound obtained with this procedure is called *precedence-based lower bound* (LBP). If the obtained lower bound is not less than the completion time for a known complete schedule, the partial schedule may be pruned.

Apart from LBP, by ignoring precedence constraints and only taking resource requirements into account a *resource-based lower bound* (LBR) can be defined. With constraints on resource availabilities and resource demands of the activities, minimum project completion duration can be computed. One way to improve the lower bounds is to incorporate heads and tails into LP-formulation (Moder et al. 1983).

Demeulemeester and Herroelen (1992) developed a branch and bound procedure for the multiple resource-constrained project scheduling problem. The procedure was based on a depth-first solution strategy in which nodes in the solution tree represent resource and precedence feasible partial schedules. Branches emanating from a parent node correspond to exhaustive and minimal combinations of activities, the delay of which resolves conflicts at each parent node.

Brucker P. et al. (1998) proposed a branch and bound algorithm for the resource-constrained project scheduling problem. The branching scheme starts from a graph representing a set of conjunctions, the classical finish start precedence constraints, and disjunctions, induced by the resource constraints. The algorithm either introduces disjunctive constraints between pairs of activities or places these activities in parallel. Concepts of immediate selection are developed in connection with this branching scheme. Immediate selection allows adding conjunctions as well as further disjunctions and parallel relations. The computational tests show that the algorithm does not perform well on problems with small capacity factor (up to 0.5). To improve the performance for such problems better lower bounds should be derived.

A depth-first branch and bound procedure proposed by Speranza and Vercellis (1993), and exact branch-and-bound algorithm (Sprecher and Drexl, 1998) are also proposed.

As the early attempts at using integer programming to solve the exact version of this problem were unsuccessful, numerous specialized approaches for scheduling resource constrained project scheduling problem were developed (Johnson 1967, Davis 1969, Balas 1970 and 1971, Schrage 1970, Davis and Heidorn 1971, Gorenstein 1972, Fisher 1973, Patterson 1973, Patterson and Huber 1974, Patterson and Roth 1976, Talbot 1976, Stinson 1976, 1978, Patterson J. H. 1984, Kurtulus and Narula 1985, Norbis and Smith 1988).

The studies on heuristic algorithms has shown that one heuristic which gives good results for a project might not give that much successful results for another project. This is the greatest disadvantage of the heuristics: Rules perform well on one problem may perform poorly on another. In practice, even with more sophisticated procedures, it is not possible to guarantee in which particular heuristic, or combination of heuristics, will produce best results for a given problem. In spite of not guarantying finding the global optima, heuristic algorithms are used widely for the solution of SRCPSP. Some of the important heuristics are illustrated in Table 2-1.

In the optimization of manufacturing projects, obtaining global optima is certainly essential. However, when the construction projects are concerned, the schedules obtained by heuristic algorithms are acceptable to use for construction planning purposes when the uncertainties of the activity durations, resource constraints and labor productivity are considered (Moder et al. 1983).

Comparison of the heuristic algorithms is performed by Davis and Patterson (1975). In the analysis, small and medium sized multi-resource problems for which the optimal solution, in terms of minimum project completion duration, could be calculated. Deviation of the heuristic rules from the optimum duration is given in Table 2-2. The explanations of the heuristic algorithms can be found in Table 2-1.

Table 2-1 Some of heuristics used for the solution of SRCPSP

Heuristic Scheduling Rules Evaluated		
Rule	**Notation**	**Operating Features**
Minimum activity slack	MINSLK	Schedules first those activities with lowest activity slack time (total float)
Minimum late finish time	LFT	Schedules first those activities with the earliest values of late finish time
Resource scheduling method	RSM	Priority index calculated on basis of pairwise comparison of activity early finish and late start times. Method gives priority to activities roughly in order of increasing late finish time
Greatest resource demand	GRD	Schedules first those activities with greatest resource demand in order to complete potential bottleneck activities
Greatest resource utilization	GRU	Gives priority to that group of activities which results in the minimum amount of idle resources in each scheduling interval, involves an integer linear programming logarithm
Shortest Imminent Operations	SIO	Schedules first those activities with shortest durations in an attempt to complete the greatest number of activities within a given time span
Most Jobs Possible	MJP	Gives priority to the largest possible group of jobs which can be scheduled in an interval. Involves an integer linear programming logarithm
Random activity selection	RAN	Priority given to jobs selected at random, subject to resource availability limits.

Table 2-2 Heuristic algorithms result (Davis and Patterson 1975)

Heuristic Algorithm	**Deviation (%)**	**Optimum found (%)**
MINSLK	5,6	29
LFT	6,7	20
RSM	6,8	14
RAN	11,4	5
GRU	13,1	2
GRD	13,1	13
SIO	15,3	1
MJP	16,0	2

Recent comparison of heuristic methods is performed by Kanit et al. (2009). Ten housing projects in Turkey are investigated by three heuristic algorithms; maximum remaining path length (MRPL), LFT and MINSLCK. The performance of each priority rule is evaluated in relation to the duration of the project. Besides labor and equipment constraints, limitations in delivery of material if also considered. In the analysis MRPL priority rule present the best results.

Methods implemented for the solution of resource constrained scheduling problem includes genetic algorithms, tabu search, simulated annealing, scatter search, ant systems, mathematical formulations and linear integer programming.

GA has received considerable attention regarding their potential as an optimization technique. When using GA, chromosome patterns depend on the problem to be coded. There are two basic chromosome formats in GA: (1) binary coding; and (2) ordering coding. The forms of crossover and mutation operators also depend on the way the problem is coded. Ordering coding may represent the priority of the activities or the activities with the certain priority.

Research has been done in the optimization of construction scheduling using GA (Chan et al. 1996; Chua et al. 1997; Feng et al. 1997). Leu and Yang (1999) proposed a genetic algorithm based multi-criteria computational optimum scheduling model, which integrates the time/cost trade-off model, resource-limited model, and resource leveling model.

In addition to the researchers mentioned above, GA is also implemented by Valls et al. (2003), Kochetov and Stolyar (2003), Alcaraz et al. (2004), Hartmann (2002), Coelho and Tavares (2003), Leon and Ramamoorthy (1995) for the solution of SRCPSP. The algorithms differ in execution of the scheduling and sampling operators. Kim and Ellis (2008) implemented a GA based algorithm and tested on 15 of the projects of the PSPLIB project set.

Tabu search is implemented by Klein (2000), Nonobe and Ibaraki (2002) and Baar et al. (1998). Bouleiman and Lecocq (2003) preferred simulated annealing, while Debels et al. (2006) used scatter search. Heuristic sampling algorithm is selected by Tormos and Lova (2001, 2003a, 2003b), Valls et al. (2005), Schirmer (2000), Kolisch (1996a, 1996b, 1995), and Kolisch and Drexl (1996).

Ant colony optimization has recently been implemented for the solution of SRCPSP. Stützle (1998), Merkle et al. (2002), Gagne et al. (2002), Besten et al. (2000), Blum and Sampels (2004), Blum (2005a) are the primary studies on the SRCPSP with the search of optimum by ACO. Merkle et al (2002) tested the algorithm on the test problems of PSPLIB and the algorithms could not converge into the schedules better than 30% on the average from the lower bounds of the 120-activity problem set.

Resource constrained scheduling problem is still an up-to-date subject and there are many recent studies. Tseng and Chen (2006) developed a hybrid meta-heuristic algorithm called ANGEL which is the combination of ACO and GA. Initially, ACO searches the solution space and generates activity lists to provide the initial population for GA. After this step, GA is executed and the pheromone set in ACO is updated if GA obtains better solution. This forms a cycle of the meta-heuristic method ANGEL and this cycle is iterated until the stopping criterion is met. The meta-heuristic algorithm ANGEL obtains successful results in the analysis of the 30-activity test set, however the analysis results are not satisfactory in the 120-activity test set.

Wnag and Zheng (2001), developed a hybrid meta-heuristic algorithm by combining genetic algorithm and simulated annealing. With the hybrid meta-heuristic algorithm, some benchmark job-shop scheduling problems are well solved by the hybrid optimization strategy, and competitive results are obtained with the best literature.

Mendes et al. (2009) generated a GA based on binary representation. The schedule is constructed using a heuristic priority rule in which the priorities of the activities are defined by the genes of the GA. Each chromosome is made of genes whose amount is twice of the activity number of the project. The genes carry information related with the activity priority and delay time for the activity. Mendes et al. (2009) obtained the most successful results in the analysis of 30, 60 and 120-activity test sets of PSPLIB.

Ranjbar (2008) proposed a new heuristic algorithm for the SRCPSP based on filter and fan (F&F) method. The proposed method incorporates two fundamental components. First component is a local search to identify a local optimum and a F&F search to explore larger neighbors to overcome local optimality. If a new local optimum is found, the method switches the search strategy. The switching continues until the filter and fan search fails to improve the current best solution. Ranjbar (2008) obtained very close results to the current best analysis in 30 and 60-activity project data sets, the algorithm is slightly far from the current best values in 120-activity projects.

Based on the recent studies on SRCPSP followings can be concluded, LP requires solution of high amounts of equations and parameters even for small sized problems. B&B algorithm may require large amounts of memory capacity for the storage of partial networks. Heuristic algorithms are problem based solution algorithms whose results may deviate significantly for different network characteristics. Meta-heuristic algorithms require too much evaluation in order to converge into the near-optimum solutions. It is seen that the convergence speed of the meta-heuristic algorithms should be improved in order to obtain proper results for the solution of RCPSP.

2.5 Multi Mode Resource Constrained Project Scheduling

In the classical SRCPSP each activity has a single execution mode, that is, both the activity duration and its resource requirements are fixed. In consideration of the dynamic availabilities of the renewable and non-renewable resources as well as the dynamic precedence feasibilities, some activities are allowed to be executed in flexible forms or modes in practice; each mode of an activity represents an alternative relation between resource requirements of the activity and its corresponding duration. Therefore, the multimode resource-constrained project scheduling problem (MRCPSP) that allows each activity to be executed in one of several modes needs to be addressed.

MRCPSP considers both renewable and nonrenewable resources that have not been addressed efficiently in the construction field. Zhang et al. (2006) introduces a methodology for solving the MRCPSP based on particle swarm optimization (PSO). A particle representation formulation is proposed to represent the potential solution to the MRCPSP in terms of priority combination and mode combination for activities.

The methodologies for solving the MRCPSP that have been proposed include the exact and heuristic or meta-heuristic approaches. The exact methods include the exact enumeration schemes proposed by Talbot (1982) and Patterson et al. (1989). Enumeration scheme based on the enumeration scheme for single mode (Sprecher et al., 1997) and (Demeulemeester and Herroelen, 1992) is also proposed. The heuristic methods include the branch-and-bound algorithm that is developed based on the exact procedure (Hartmann and Drexl, 1998), the biased random sampling approach (Drexl and Grünewald, 1993), the single-pass and a multi-pass approach (Slowinski et al., 1994), the local search procedure (Kolisch and Drexl, 1997), a heuristic algorithm based on a new mathematical formulation (Maniezzo and Mingozzi, 1999), and one based on the CPM computation (Boctor, 1996).

Peteghem and Vanhoucke (2009) present a detailed description of the past studies on the MRCPSP. The studies are classified as exact, heuristic, meta-heuristics algoritms.

The PSPLIB problems are solved by the following authors. Kolish and Drexl (1997) analyzed J10 and J30 activity sets with heuristic algorithms. Hartmann and Drexl (1998) analyzed J10, J12, J14 and J16 problem sets by B&B algorithm. Sprecher and Drexl (1998) implemented B&B algorithm using his own data set apart from J10, J12, J14, J16, J18 and J20 data sets. Özdamar (1999) used his own 90-activity multi-mode problem sets and the J10 problem set of the PSPLIB. Nonobe and Ibaraki (2001) analyzed the J30 problem set by Tabu Search (TS) algorithm. Jozefowska et al. (2001) examined J10, J12, J14, J16, J18, J20 and J30 data sets by simulated annealing algorithm. The same problem sets are analyzed by Hartmann (2001) using GA, Bouleimen and Lecocq (2003) using SA, Alcaraz et al. (2003) using GA. Zhang et al. (2006) analyzed the same problem set by ignoring J30 data set with PSO. Zhu et al. (2006) analyzed the J20 and J30 data sets by B&C algorithm. Pourghaderi et al. (2008), proposed a scatter search algorithm for the MRCPSP. Jarboui et al. (2008) implemented PSO and tested his algorithm by solving all JXX problem sets. Ranjbar et al. (2009) solved the same problem set by only ignoring J30 with scatter search algorithm. Lova et al. (2009) analyzed the all JXX problem set by GA.

Agarwal et al. (2007) analyzed MRCPSP with artificial immune system (AIS) meta-heuristic algorithm. AIS is defined as an abstract ormetamorphic computational system based on the ideas inferred from the theories and components of immunology. The algorithm involves *immune cell* and *antigens*. B-cells and T-cells are the immune cells which help in recognizing antigenic patterns. Antigens are disease causing elements consisting of self and non-self cells. Non-self antigens disease-causing and self antigens are harmless.

Hyper-mutation and receptor editing are two remarkable characteristics of the AIS. Hyper-mutation is very much similar to the mutation operator of GA where hypermutation differs in the rate of modification which depends on antigenic affinity. Lower antigenic affinity antibodies are hyper-mutated at a higher rate as compared to the antibodies with higher antigenic affinity. This phenomenon is known as receptor editing, which governs the hyper-mutation. Hyper-mutation is an optimum search operator while receptor editing avoids being stuck into local optima.

Damak et al. (2009) implemented Differential Evaluation (DE) meta-heuristic algorithm and obtained successful results. DE is inspired by GA and the evolutionary strategies. However, DE is combined with geometric search technique. The key idea behind DE is a scheme for generating trial parameter vectors. Mutation and crossover are used to generate new vectors, and selection determines which of the vectors will survive the next generation.

Tchao and Martins (2008), implemented tabu search based heuristic algorithms with path relinking for the MRCPSP. Path relinking is used as a post optimization strategy, so that it explores paths that connect elite solutions found by the tabu search based heuristics. The combined hybrid heuristics were able to find near-optimum solutions in quite short computational times.

Recent analysis results of the researches on the multimode problem sets of Kolisch and Sprecher (1996) are given in Table 2-3. Some of the analyses do not include the J30 problem set. In addition to this, there is not a well defined stopping criteria of the schedules. As a result of this, the final obtained average deviation and percentage of optimum solutions obtained could not be compared fairly. Table 2-4 contains analysis results of the same problem which are fairly older analysis.

Table 2-3 Analysis results of recent previous studies

	Zhang et al. (2006)		Jarbouri et al. (2008)		Jarboui et al. (2008)		Porghaderi et al., (2008)	
	PSO		PSO		Differential Evaluation		Scatter Search	
Prob Set	Av Dev	Per Op. Found	Av Dev	Av Dev	Av Dev	Per Op. Found	Av Dev	Per Op. Found
J10	0.11	97.9	0.03	99.25	0.06	98.7	0.05	99.07
J12	0.17	95.2	0.09	98.47	0.14	97.3	0.11	98.35
J14	0.41	89.3	0.36	91.11	0.44	89.8	0.38	93.66
J16	0.83	85.6	0.44	85.91	0.59	87.8	0.52	87.66
J18	1.33	74.5	0.89	79.89	0.99	78.3	0.84	83.33
J20	1.79	69.1	1.10	74.19	1.21	73.3	1.03	79.24
J30			2.35	57.41			1.76	69.44

Table 2-4 Analysis results of the previous studies

	Sprecher and Drexl (1998)		Hartmann, (2001)		Bouleimen and Lecocq, (1998)	
	B&B		Simulated Annealing		Genetic algorithm	
Prob Set	Av Dev	Per Op. Found	Av Dev	Per Op. Found	Av Dev	Per Op. Found
J10	0.0	100	0.21	96.3	0.06	98.7
J12	0.12	98.2	0.19	91.2	0.14	97.3
J14	1.46	85.7	0.92	82.6	0.44	89.8
J16	3.81	69.5	1.43	72.8	0.59	87.8
J18	7.48	57.4	1.85	69.4	0.99	78.3
J20	11.51	47.3	2.10	66.9	1.21	73.3

Mori and Tseng (1997), proposed a GA based solution algorithm and obtained better results than stochastic scheduling method of Drexl and Gruenewald (1993). Goçalves et al. (2008) proposed a GA algorithm for the solution of multi-mode multi projects. The gene representation contains activity priorities, delay for each activity and release date of the project. The binary representation for the gene representation is preferred.

2.6 Resource Constrained Time Cost Trade-off Problem

The final problem type examined in this thesis is the resource constrained time cost trade-off problem (RCTCTP). This problem is very similar to simple TCT and MRCPSP. It differs from simple TCT in a way that there are limitations on resource availabilities and differs from MRCPCP that the objective function to be minimized is the cost of the project not the duration of the project.

Compared by the TCT and MRCPSP, optimum solution of RCTCTP gives the most profitable project schedule. In this case, cost of the project is minimized by considering the indirect costs and the liquidated damages. On the other hand, minimum project duration minimizes indirect project costs which may not minimize the total project cost. Solution of simple TCT problem may end up with unacceptably fluctuated and peaked resource profile, as the resource demand is never considered during the analysis. The fluctuated resource profile may cause excessive decrease in the labor productivity. In addition to this, significant number of labor and equipment may stay idle which may also cause an increase in the direct cost of the project. As a result of this, the project may cost more than the planned project by simple TCT analysis.

RCTCTP is significantly a difficult problem not only to solve but also difficult to acquire the necessary data for the analysis. Apart from the duration and resource requirements, the analysis also demands the costs of each execution mode of the activities. As a result of this, the data required for the analysis becomes time-consuming and requires detailed planning endeavor. If manufacturing is considered, there would be enough time, work force and money for the data acquisition. However, construction industry usually prepares planning data in a short time and limited budget. As a result of this, there would be uncertainties on the data which will affect the benefits of the data.

In order to prepare dependable project plans, construction companies should allocate more budgets and implement computer aided tools for preparation of the tender documents. With the help of a database and a decision support system, costs of the construction activities can be estimated in a short time and accurately.

As it is a difficult problem and the data required for the analysis is difficult to acquire the RCTCTP did not attract the attention of many researchers. Pathak et al. (2007) analyzed RCTCTP by using a GA based method. The optimum solution is searched by multi-objective genetic algorithm (MOGA). For the test of the analysis 9-activity and 18-activity projects are used. Adaptive resource restrictions are defined during the project which aims to minimize the maximum resource demand.

RCTCTP is analyzed by dividing the problem into two separate problems in which the simple TCT problem is analyzed by releasing the resource constraints and the obtained schedule is resource leveled. However, this procedure may not always help reducing the peak resource demand. As a result of this infeasible labor or machinery demand may be faced during the life cycle of the project.

As RCTCTP is not widely analyzed in the literature, there are not test project sets available. As a result of this, test projects are generated to be used in the RCTCTP analysis. The test problems will be made available after the dissertation of this thesis study.

CHAPTER 3

META-HEURISTIC ALGORITHMS

In this thesis study, meta-heuristic algorithms are developed for the solution of optimization problems which are discussed in Chapter 2. Some of the existing meta-heuristic algorithms are implemented without any modification, some algorithms are modified and their convergence capability is improved and new meta-heuristic algorithms are developed by embedding meta-heuristic algorithms into Genetic Algorithm. In this chapter, meta-heuristic algorithms implemented for this study are explained and their executions are illustrated.

A meta-heuristic is an optimum search algorithm which is independent from problem type where heuristic algorithms are problem dependent. The name combines the Greek prefix "meta" ("beyond", here in the sense of "higher level") and "heuristic" (from ευρισκειν, heuriskein, "to find") (Reeves 1995a).

3.1 Genetic Algorithm

Evolutionary computing was introduced in the 1960s by I. Rechenberg in the work “Evolution strategies”. This idea is later improved by researchers and finally, (GA) was invented by John Holland and developed this idea in his book “Adaptation in natural and artificial systems” in the year 1975. Holland proposed GA as a heuristic method based on “Survival of the fittest”. GA was discovered as a useful tool for search and optimization problem.

GA is a search technique used for finding exact or near optimum solutions to optimization problems. GA searches the global optimum with an algorithm based on the meiosis. An initial population is randomly generated and new genes are reproduced by crossover. The genetic differences are formed by mutation and the unfit genes are terminated by natural selection operations.

As its name implies, each solution candidate is represented by genes in GA. For the TCT type problems, the gene contains the crashing option assigned to that activity among possible alternatives. For the resource leveling problems genes contain the delay time of the activities which have floats, and for the resource allocation genes represent activity priorities. Gene representation can be integer, float, binary or hexadecimal. Binary representation has advantages in crossover and mutation operations, because of this binary representation is preferred for the solution of simple TCT and resource leveling problems although it increases computational demand.

First step of the GA is generation of the initial population. Determining the population size has significant importance, because small populations contain the risk of seriously under-covering the solution space, while large populations incur severe computational demand. Goldberg indicates that the optimal size for binary-coded strings grows exponentially with the length of the string *n* (Reeves 1995a).

Population generation can be executed randomly or by seeding. Seeding is generating the initial population by high-quality solution obtained from another heuristic technique. Seeding can help GA finding better solutions rather more quickly than it can do from a random start. However, there is a possible disadvantage in that the chance of premature convergence may be increased (Revees 1992 and Kapsalis et al. 1993).

Crossover is the necessary operation for the genetic reproduction. New genes are reproduced from randomly selected genes. Couples, namely the parents; are determined by randomly generated numbers and new two genes are reproduced from parents by crossover operation. The location of the crossover is also determined by generating a random number which is shown in Figure 3.1. After the crossover new two gene combinations are generated by the existing gene combination of the population.

Eshelman et al. (1989), worked on multipoint crossover that examined the biasing effect of traditional one-point crossover and considered a range of alternatives. Central argument was that two sources of bias exist to be exploited in a genetic algorithm; positional bias, and distributional bias. Eshelman concluded that simple crossover has considerable positional bias and the bias may be against the production of good solutions. In addition to this, crossover operator is analyzed in detail by Faily (1991).

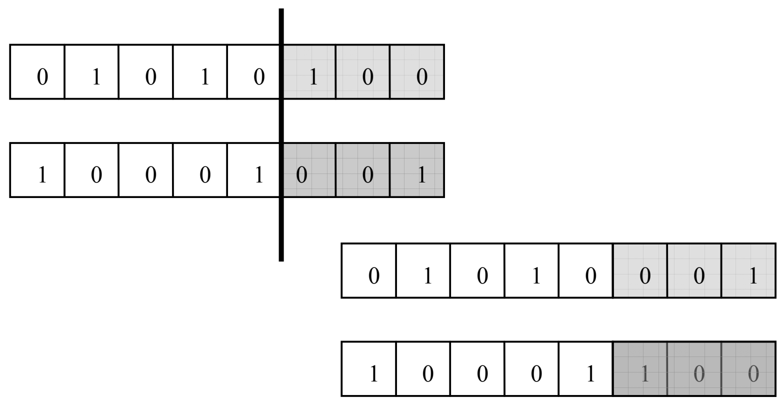

Figure 3.1 Crossover operator

To examine the results of multiple point crossover operations, apart from one point crossover; two, three and four point crossover operations are also examined. In multiple point type of crossover operations, the gene is divided into equal intervals, which is same with the crossover point number. For each interval simple crossover operator is executed. Effect of chromosome number is analyzed on 18-activity project and the analysis results are given in the next chapter.

In Holland's original GA, parents were selected by means of a stochastic procedure from the population and a complete new population of offspring was generated which replace their parents. In another version, he suggested that each offspring should replace a randomly chosen member of the current population.

In this study, in order to select the genes for the crossover operation, one random number is generated for each gene. The genes, which come up with random numbers smaller than the crossover probability are selected for the crossover. The couples are formed by sorting the selected genes according to their assigned random numbers and each consecutive genes form couple. With this method both selection of genes for the crossover and the determining crossover couples will be performed randomly. Mutation operator shifts the binary value of the gene on a randomly selected location from 0 to 1 or vice versa, which is shown in Figure 3.2.

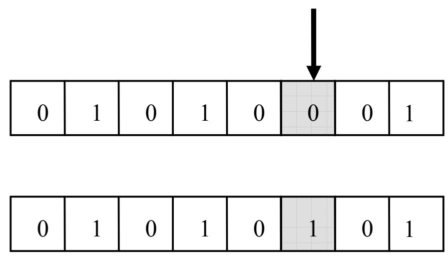

Figure 3.2 Mutation operator

Mutation prevents domination of a certain gene which has high probability of survival. Initially domination of relatively good fit genes may cause being stuck into local minimum. On the other hand, too high mutation rate may also bastardize good fit genes. Moreover, crossover can produce good fit genes from existing genes, but it can not generate a new gene for a specific portion which does not exist in the population. Therefore, mutation operator has significant importance as it can produce new gene combinations, which have not been generated at the initialization of the population or regenerate a gene combination terminated at natural selection.

Crossover is more important in the beginning when the population is diverse, but as the individuals approaches to optimum solution it is important to increase the chance of finding different solution, which is where mutation is more effective.

Natural selection is the final step of a cycle of the GA. Natural selection keeps the population size constant by terminating the same number of individuals reproduced at the crossover. In addition to this, it improves the overall gene quality of the population by terminating the low fit genes. On the other hand, low fit genes may carry very important genes on their certain location and in order to preserve these portions and prevent initially good fit genes to dominate, some precautions are taken at the natural selection phase. Roulette wheel selection algorithm has been implemented for this purpose which is a probabilistic selection algorithm. Roulette wheel determines the genes to be terminated by assigning high probability of termination to low fit genes and low probability of termination to good fit genes.

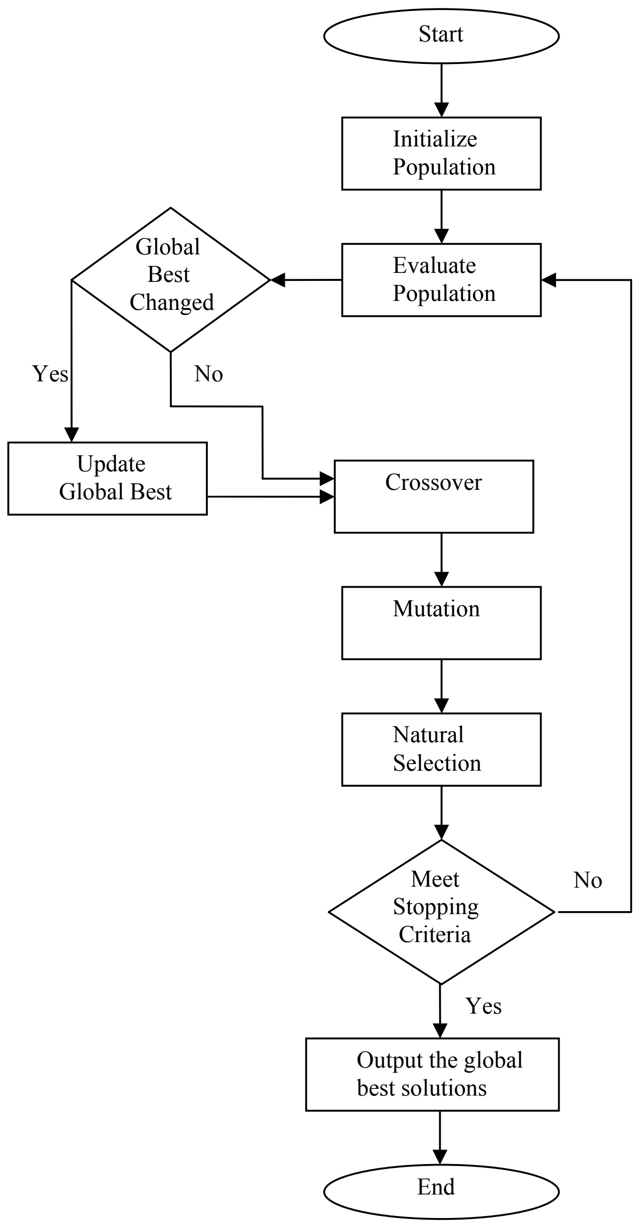

Figure 3.3 Flowchart of GA

An obvious defect with the simple natural selection operator is that there is no guarantee that best member of a population will survive in the next generation. One way of dealing with this drawback is De Jong’s (1975) elitist model, in which the best member of the current population is forced to be a member of the next. Another approach was taken by Ackley (1987), who introduced the term “termination with prejudice” for his algorithm. This was to use an incremental replacement approach where at each step the new chromosome is replaced by a randomly selected one from these which currently have a below average fitness. With this algorithm good fit genes are guaranteed to survive if they are not terminated by mutation.

Fitness calculation has also significant importance, since a problem may arise in the beginning of the GA when there are many poor chromosomes and just one or two outstanding chromosomes. A naive fitness measure may lead to a rapid takeover by the good fit ones and premature convergence to a poor local optimum can be unavoidable. One approach to prevent this is to ignore the actual objective function and use a ranking procedure. The main argument is that the key to good GA performance is to maintain an adequate selective pressure by means of an appropriate relative fitness measure.

Natural selection operator completes the one cycle of the GA. Number of cycle generation depends on the number of input parameters and the expected reduction in the total project cost. Flowchart of GA is given in Figure 3.3.

3.2 Genetic Algorithm with Simulated Annealing

If GA is implemented solely for the optimization, much iteration would be required to obtain satisfactory results. Convergence of GA can be increased significantly by applying complementary methods, thus important savings would be obtained in terms of computation time. Simulated Annealing (SA) is one of the complementary methods that are used for this purpose. SA is a generic probabilistic meta-heuristic algorithm for the global optimization problem. SA is inspired by the cooling schedule of alloys subjected to tempering. Initially, when the temperature is high, the molecules are free to move in any direction. At later phases, movements of molecules are restricted depending on the temperature (Reeves 1995b).

Mutation operator sometimes leads to better genes and sometimes doesn't. SA decides weather to reject or accept the mutation that leads to a worse result. The rejection probability increases as the iteration number increases which simulates the cooling of the alloy. SA accepts every mutation that leads to a better gene and decides the rejection of a harmful mutation. If the final solution is to be independent from the starting solution, the initial temperature must be hot enough to allow an almost free exchange of neighbouring solutions. In some cases there is enough information in the problem to estimate the size of the permissible increase in the cost function. If the maximum difference of the evaluation function between neighbouring solutions is known, it may be assumed that increases of cost in this magnitude will be sufficient and t can be calculated appropriately. However, computing the difference between neighbours is very time consuming in combinatorial problems. For large networks, defining the initial temperature by this way will cause too much computational demand.

Besides the initial temperature, the cooling schedule has vital importance as well. In theory, the temperature should be allowed to decrease to zero before the stopping condition is satisfied. However, in practice there is no need to decrease the temperature this far. Given the limited precision of any computer implementation, as t approaches zero from right, probability of accepting a harmful mutation will be indistinguishable to zero. Even before zero temperature is reached, it is likely that the chances of a complete escape from the current local optimum will become negligible. Thus the criterion for stopping can be expressed either in terms of a minimum value of the temperature parameter, or in terms of the 'freezing' of the system at the current solution.

Starting the process with the temperature so high that almost all mutations are accepted simply produces a series of random solutions, each might be a starting solution. In this case, any beneficial changes might be terminated by the opposite mutation because of the very high temperature and very little progress would be obtained after several iterations.

If the initial temperature is not high enough or cooled very rapidly, there can be no beneficial mutations after a certain point. If no progress is apparent in searching, a concerted acceptance of detrimental mutation would be made in order to widen the scope of the search. Kirkpatrick et al. (1983) proposed reheating the temperature if there is not an improvement for a certain number of iterations. In this thesis study, there is not any reheating, by enlarging population size; enrichment of the gene content is aimed to be obtained.

In this study, the difference between the mutated gene evaluation and the initial gene evaluation is normalized by dividing the difference by the initial evaluation value. The cooling process is controlled by Boltzmann Constant which is taken as 1 for GA. Division by temperature for cooling is replaced by multiplying the exponential equation with the iteration number. After the mutation, a random number is generated for the decision and if the generated random number is smaller than the decision function, the mutation is accepted (Hwang and He 2006). The decision function explained above is represented as:

$$Decision \quad \begin{cases} accepted \ \ if \ \ R_n \leq e^{\frac{(f_m - f_0)t}{f_0 BC}} \\ rejected \ \ if \ \ R_n > e^{\frac{(f_m - f_0)t}{f_0 BC}} \end{cases} \tag{3.1}$$

where;

R_n is a random number generated between 0 and 1 for the decision, f_m is the evaluation value of the mutated gene, f_0 is the initial value of the gene before the mutation operator affects the gene, BC is the Boltzmann constant used to determine the speed of cooling, t is the current number of iteration. In Figure 3.4 the acceptance probability of a harmful mutation which increase the evaluation function to be minimized by 2%, project schedule is plotted for Boltzmann Constant equal to 1.0.

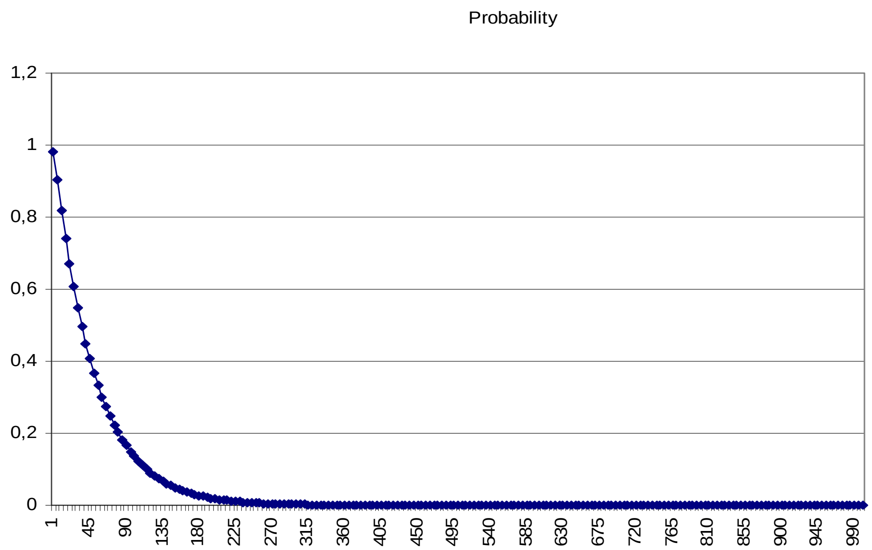

Figure 3.4 Acceptance probability vs. iteration number

As seen in Figure 3.4 after the 200th iteration, the acceptance of a harmful mutation will almost be zero. This prevents harmful mutations to take place thus mutations always seek for better neighbors and the crossover operator of the genetic algorithm prevents being stuck into local minimum.

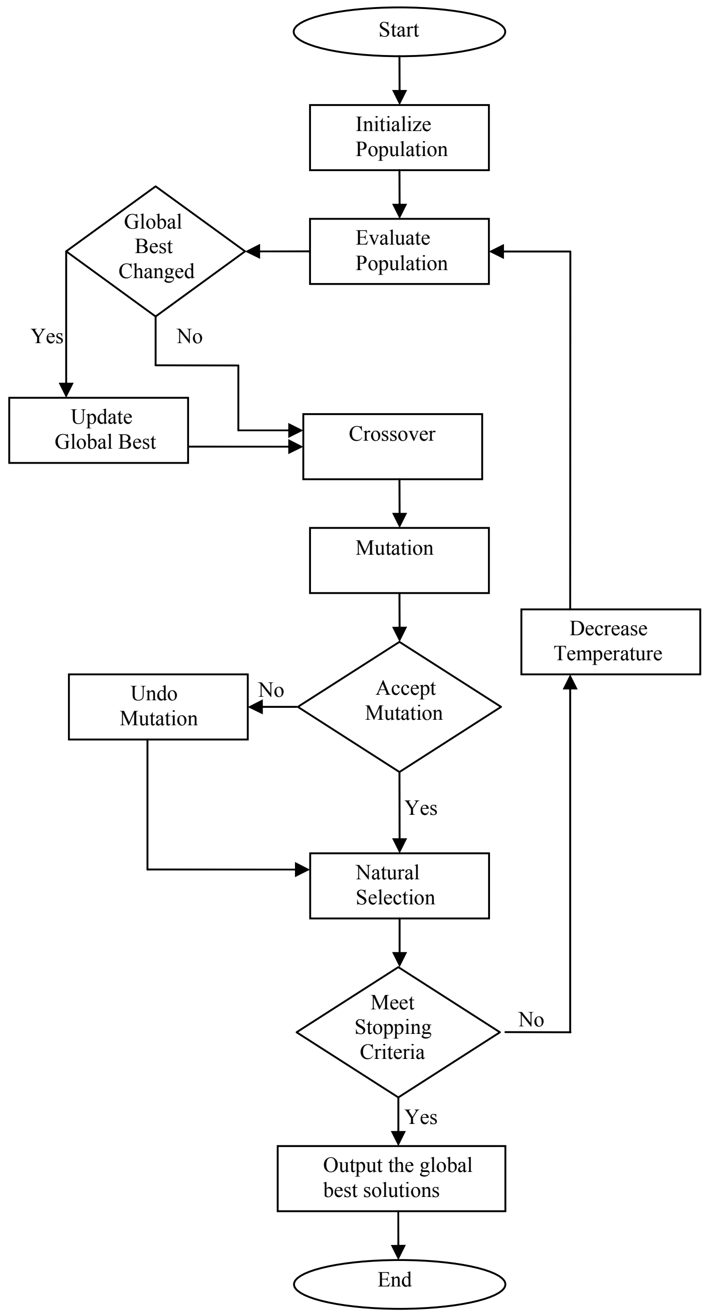

Figure 3.5 Flowchart of GASA

Decision function always gives results greater than 1 if the mutation is beneficial, as a result beneficial mutations are always accepted. If the mutated gene is worse than its initial state, the decision formula gives a result between 0 and 1 depending on the difference between the initial and mutated state. Higher the detriment of the mutation, closer the decision function to 0. If the detriment of the mutation is small the decision formula will give results close to 1 and the probability of acceptation will be high. Meanwhile, the higher the iteration number, the harder the acceptance criteria. If mutation is harmful even a small difference will be evaluated as close to 0 by the decision formula and the probability of acceptance will be very low. The hardening of acceptance criteria is controlled by the Boltzmann constant.

The genetic algorithm in which the acceptance of mutation is under the control of simulated annealing is called, Genetic Algorithm Simulated Annealing (GASA). The flowchart of GASA is given in Figure 3.5.

3.3 Hybrid Genetic Algorithm with Simulated Annealing

Population based search is ideal for exploring as much of the search space as possible. Once the GA can not find any better individuals after a certain number of generations, the best individual of the population is chosen to undergo a series of random walks until the optimal solution is found. However, one series of random walk may not be adequate for obtaining the optimum, as the number of activities of the project increases. In order to preserve the improvements gained at the previous random walks and produce better genes at the crossover, overall gene quality of the population should be improved. For this reason, the series of random walk is applied to the whole population. In the analysis of TCT and resource leveling whole population is subjected to a series of random walks. However, in resource constrained scheduling problems only population best and some of the randomly selected individuals of the population are subjected to random walk.

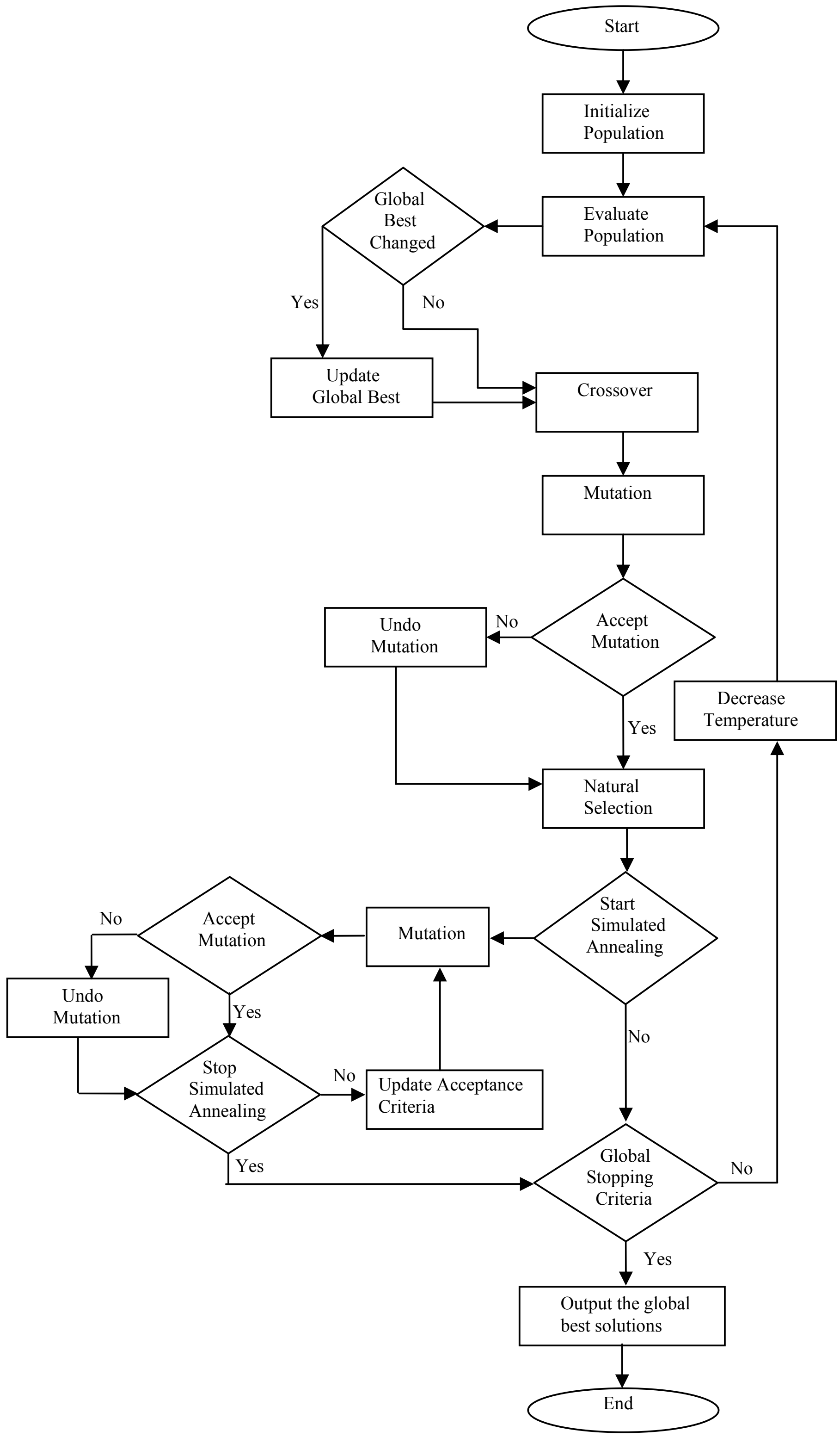

Figure 3.6 Flowchart of HGASA

The decision of applying random walk significantly increases the computational demand. However, random walk of only the best individual has high risk of getting stuck in local optimum. Therefore, in order to avoid getting stuck into local optimum hybrid local search algorithm is applied to some of the randomly selected individuals of the population. The algorithm is called Hybrid Genetic Algorithm Simulated Annealing (HGASA). After a certain number of GASA generations the whole population is subjected to a series of random walk. The number of random walk generations is determined by considering the number of parameters to be solved in the optimization procedure (Chan et al. 2005).

Acceptance of random walk is also determined by SA. However, in this case the acceptance criterion is hardened at each random walk and it is reset if another individual starts random walk process.

After the completion of random walk session, GASA starts and after a certain number of generations again random walk session starts. Number of GASA generations should be adequate enough to escape from local minimum and should not be too high to create computational burden. Hardening of the acceptance criteria in the random walk session has also important affect. Too fast cooling may cause getting stuck into local minima and too slow cooling may lead to randomized genes at the end of the random walk. Flowchart of HGASA is given in Figure 3.6.

3.4 Hybrid Genetic Algorithm with Quantum Simulated Annealing

Hybrid Genetic Algorithm with Quantum Simulated Annealing (HGAQSA) algorithm is very similar to the HGASA algorithm. Both algorithms depend on a series of GASA generations and random walk sessions. The difference is that in HGAQSA as its name implies, acceptance criteria is determined by a process analogous to quantum fluctuations. In quantum simulated annealing, a "current state" is randomly replaced by a randomly selected neighbor state, if the latter has a better value of the objective function (Das and Chakrabarti, 2005). The process is controlled by the *tunneling field strength T*, a parameter that determines the extent of the neighborhood of states explored by the method. The tunneling field is initially wide, so that the neighborhood extends over the whole search space; and is slowly reduced.

Shrinkage of neighborhood is very difficult to implement in binary representation. For this reason, the neighborhood is kept constant through the random walk session. However, the acceptance criteria are determined by the *tunneling field strength* parameter. T is assigned as the variance of the population and it is kept constant for each individual through the quantum simulated annealing (QSA) random walks.
Decision function in QSA random walk becomes;

$$Decision \quad \begin{cases} accepted\ if\ \ R_n \leq e^{\frac{(f_m - f_0)t}{T^*BC}} \\ rejected\ if\ \ R_n > e^{\frac{(f_m - f_0)t}{T^*BC}} \end{cases} \tag{3.2}$$

Where, T is the *tunneling field strength* and initially equal to $\sigma_{Population}$. After each iteration T is decreased by formula $T = T^*k$, in which k is a real number between 0 and 1 that gradually makes the acceptance criteria harder. Similar to cooling parameter, determination of k has vital importance. Flowchart of HGAQSA is shown in Figure 3.7.

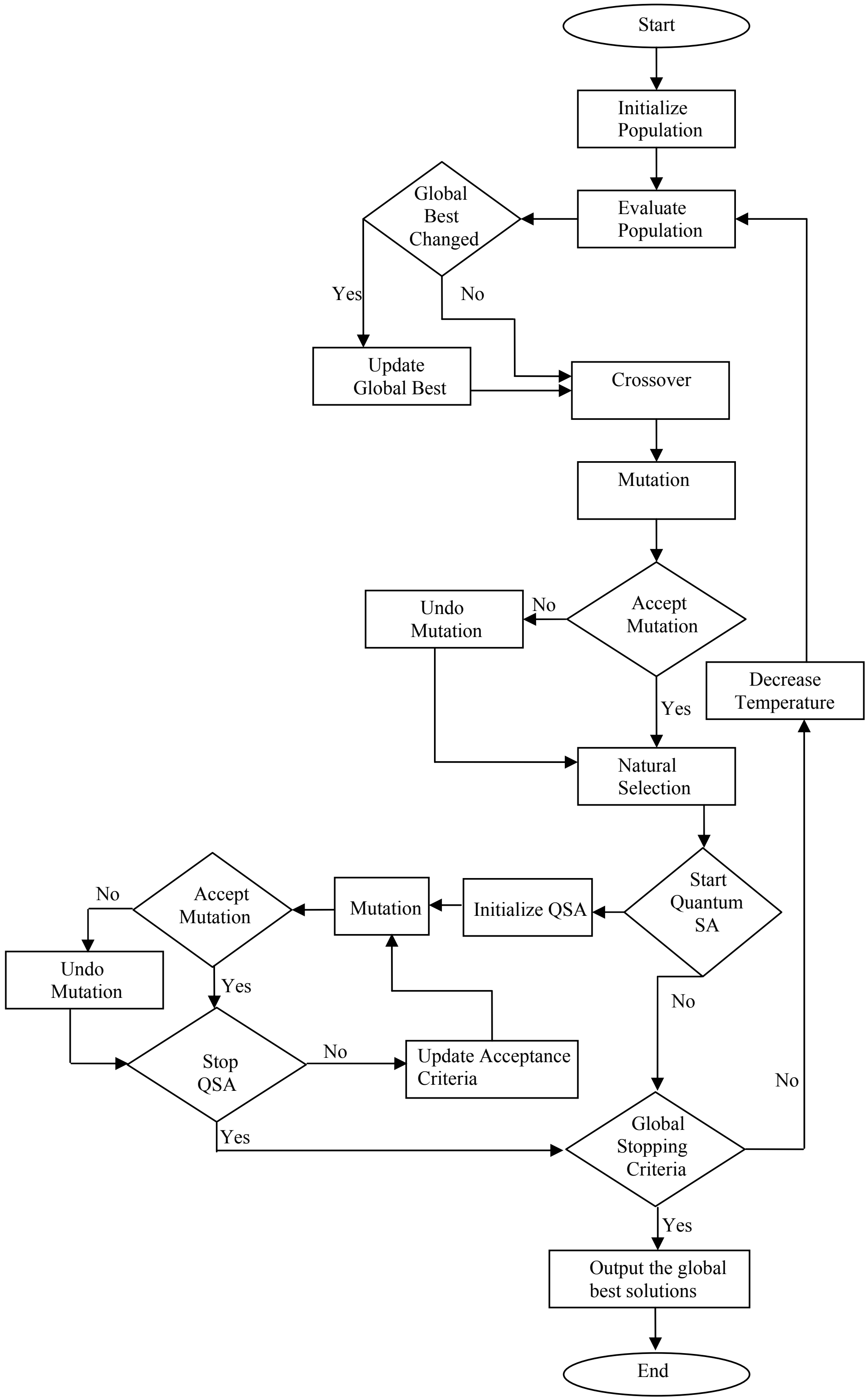

Figure 3.7 Flowchart of HGAQSA

3.5 Genetic Memetic Algorithm with Simulated Annealing

Memetic algorithm is a local search method (Merz and Freisleben 1997) and the hybrid genetic memetic algorithm simulated annealing (GMASA) is a meta-heuristic algorithm which combines the advantages of population-based search and local optimization.

In this study, a new meta-heuristic algorithm is generated by combining GA, SA and MA. Local search algorithm of GMASA differs from HGASA and HGAQSA in a way that, HGASA and HGAQSA have a random walk search algorithm while GMASA systematically searches the domain. The local search algorithm begins after a certain number of GASA generations. Memetic algorithm systematically mutates each parameter starting from the first parameter and sequentially reaches the last parameter. Acceptance decision of the mutation is given by the simulated annealing based algorithm.

After searching the whole parameter set the temperature is decreased, which means acceptance of harmful mutations are hardened, and another memetic search is performed. The number of successive memetic searches is determined according to the number of parameters and accepted mutations. If there is not any accepted mutation for a certain iteration the local search is stopped and again GASA iterations are started.

Aim of such a systematic search is to decisively visit each parameter and search its neighborhood. With the help of GASA iterations new gene combinations are formed from high quality gene combinations. After a certain number of GASA iterations, which is adequate enough to alter the gene combinations, again a set of memetic search is executed. Genetic algorithms are suitable for exploring the whole search space for identifying the possible near optimum regions. Local search operators iteratively move from one solution to a better one in its neighborhood within the near optimum region and find good solutions in small regions of the search space. Flowchart of GMASA is given in Figure 3.8.

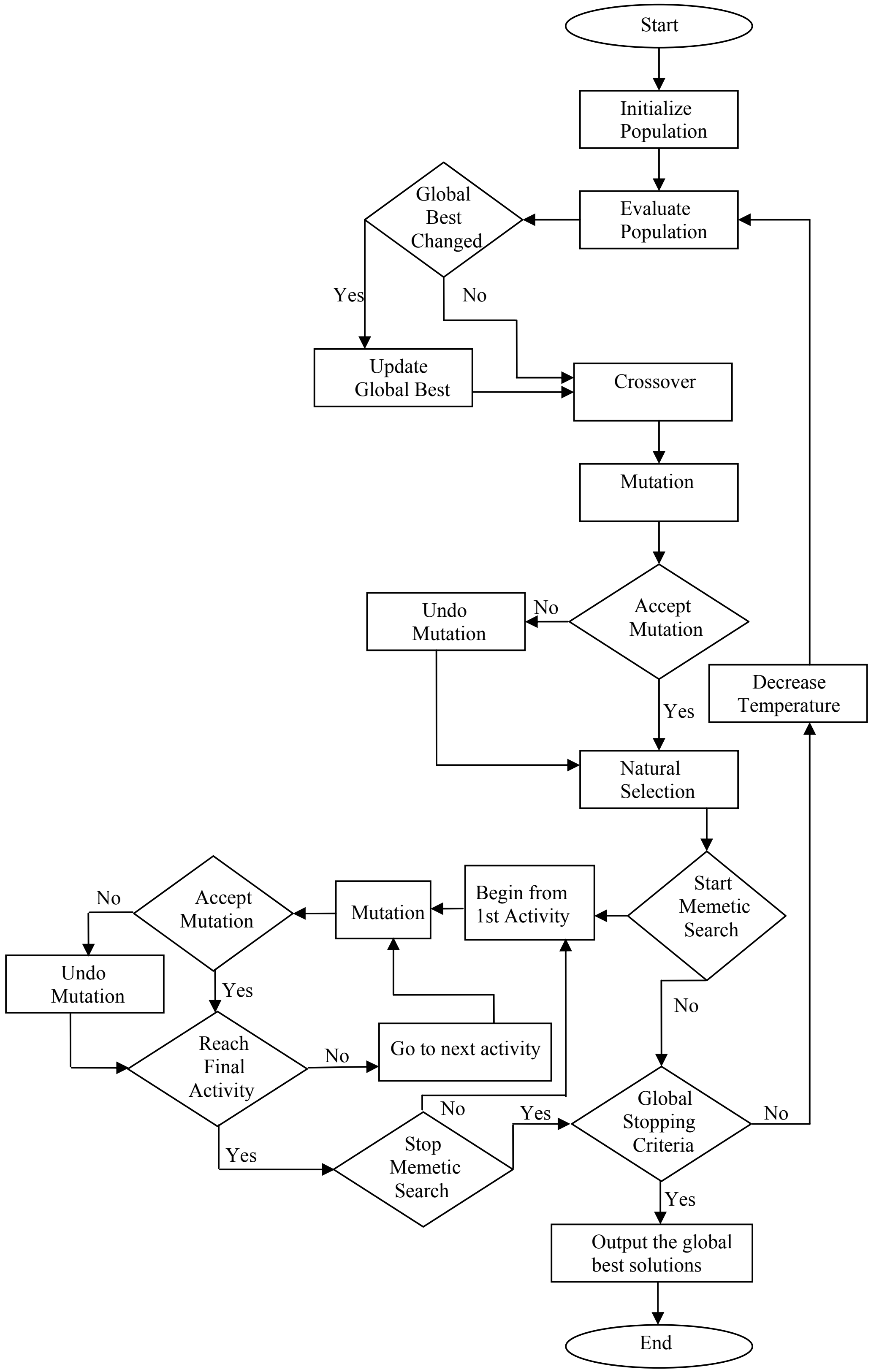

Figure 3.8 Flowchart of GMASA

3.6 Genetic Algorithm with Simulated Annealing and VNS

Genetic algorithm with simulated annealing and variable neighborhood search (GASAVNS) is a hybrid meta-heuristic algorithm which has a variable neighborhood during the local search. Variable Neighborhood Search (VNS) is a recent meta-heuristic, which exploits systematically the idea of change of neighborhood during the search (Hansen et al. 2008). Using systematically this idea, which leads to a new meta-heuristic a widely applicable algorithm is generated. Contrary to other meta-heuristics based on local search methods, VNS does not follow a trajectory but explores increasingly distant neighborhoods of the current incumbent solution, and jumps from this solution to a new one if and only if an improvement has been made (Hansen and Mladenovic 2001, Mladenovic and Hansen 1997).

GASAVNS starts with GASA and after a certain number of generations VNS algorithm is executed. VNS algorithm searches the domain with random walk search algorithm. VNS starts with one mutation per iteration and the acceptance of the mutation is decided by SA. After a several number of VNS, neighborhood is extended by increasing the number of mutations in a random walk. The maximum number of mutations in one generation is limited by a pre-determined number which is obtained by considering the number of parameters. When VNS reaches the maximum neighborhood, then VNS stops and GASA is executed.

Similar to other meta-heuristics GASAVNS has the advantages of genetic and local search algorithms. In addition to this, SA prevents obtaining a randomized solution generated by multi mutations. VNS has a fast local search algorithm, if the number of parameters is high and affect of some of the parameters are relatively low with respect to others VNS can obtain near-optimum solutions in the early stages of the iterations. In addition to this, VNS has the possibility of surviving from local optima with its multi-mutation specialty. Flowchart of GASAVNS is given in Figure 3.9.

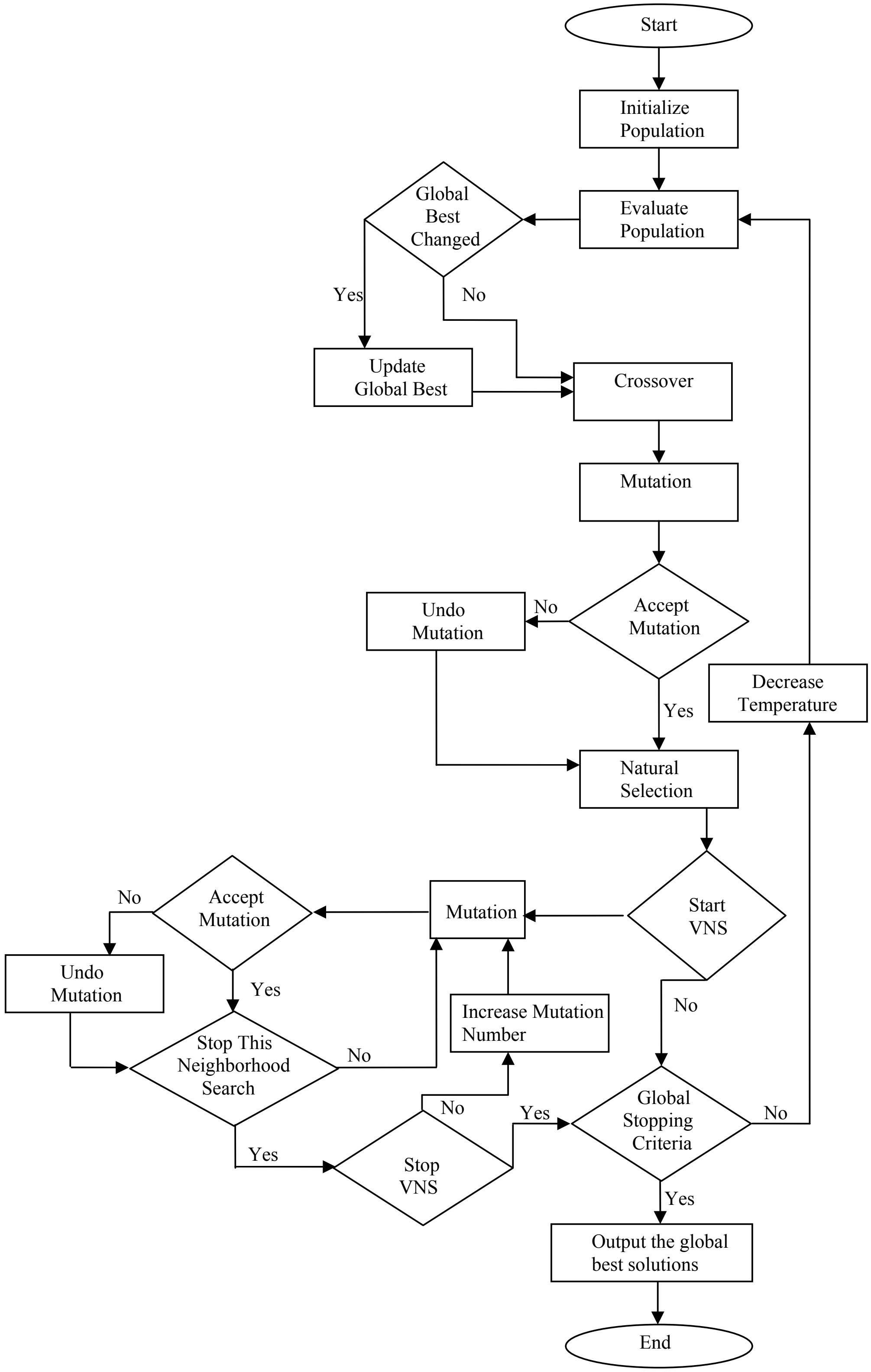

Figure 3.9 Flowchart of GASAVNS

3.7 Particle Swarm Optimization

Particle Swarm Optimization (PSO) is (Kennedy and Eberhart 1995) a meta-heuristic algorithm inspired by the social behavior of a flock of migrating birds trying to reach an unknown destination. PSO is modified by introducing a new parameter, called inertia weight by Shi and Eberhart (1998).

Similar to genetic algorithms, PSO has randomly generated population. In contrast to evolutionary algorithms, PSO simulates social behavior. Instead of using genetic operators, individuals are evolved by cooperation and competition among the individuals. Each individual adjusts its flying velocity according to its own and the population's experience. Individuals are named as *particle* which represents a potential solution to a problem.

PSO is initialized by randomly generated individuals of size N. The i^{th} particle is represented by its position as a point in a S-dimensional space, where S is the number of variables. Throughout the process, each particle i monitors three values: its current position (X_i); the best position it reached in previous cycles (P_i); its flying velocity (V_i). These three values are represented as follows:

$$
\begin{array}{ll}
\text{Current position} & X_i = (x_{i1}, x_{i2}, \ldots, x_{iS}) \\
\text{Best previous position} & P_i = (p_{i1}, p_{i2}, \ldots, p_{iS}) \\
\text{Flying Velocity} & V_i = (v_{i1}, v_{i2}, \ldots, v_{iS})
\end{array}
\tag{3.3}
$$

In each cycle, the position P_g, of the best particle, g, is calculated as the best fitness of all particles. Accordingly, each particle updates its velocity V_i to converge into the best particle g, as fallows:

$$V_{i+1} = \omega \times V_i + c_1 \times rand1(\ ) \times (P_i - X_i) + c_2 \times rand2(\ ) \times (P_g - X_i) \tag{3.4}$$

where, $V_{\max} \geq V_{i+1} \geq -V_{\min}$, $V_{\max}$ is the maximum allowable velocity, $V_{\min}$ is the minimum allowable velocity, c_1 and c_2 are two positive constants called learning factors which are usually assigned $c_1 = c_2 = 2$; *rand1( )* and *rand2( )* are random numbers in the range [0,1]; ω is an inertia weight employed as an improvement to control the impact of the previous history of velocities on the current velocity. The operator ω plays the role of balancing the global search and the local search; which is proposed to decrease linearly as iteration progresses from 1.4 to 0.5 (Shi and Eberhart 1998).

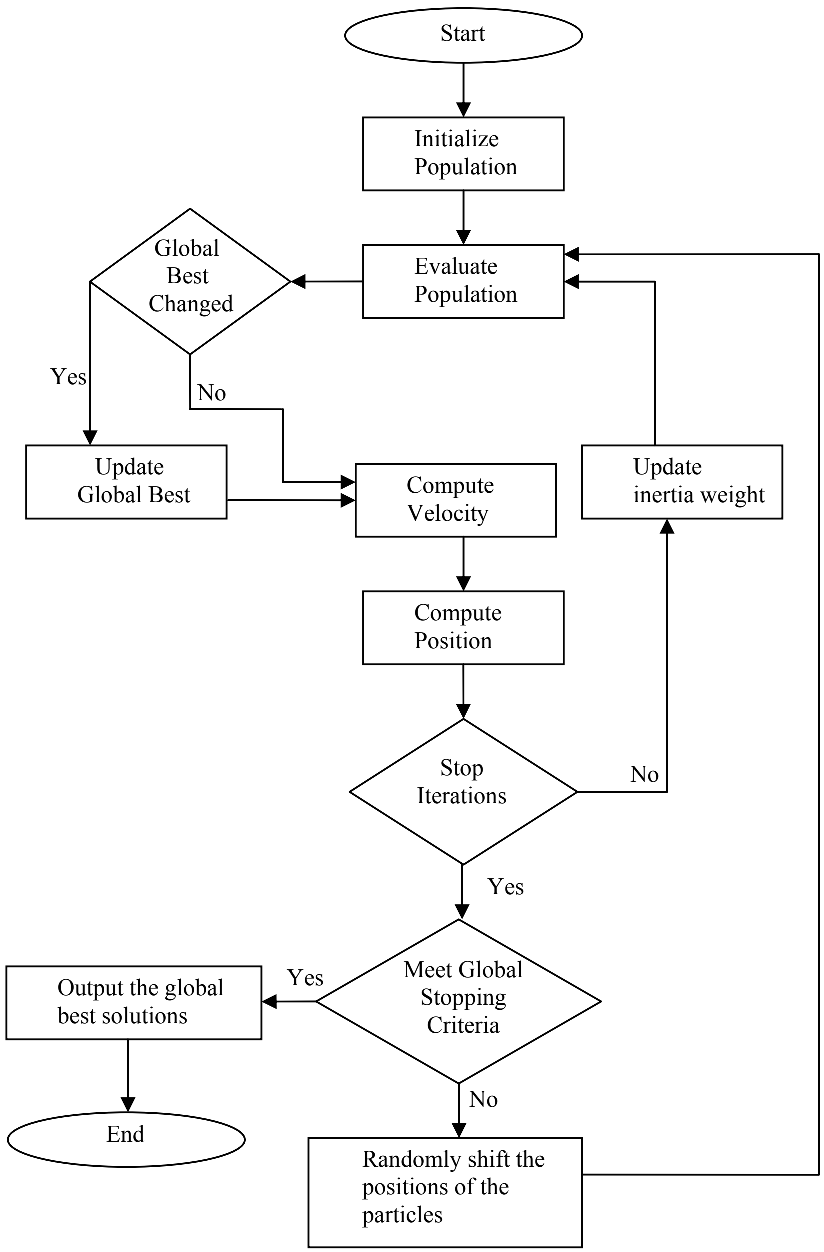

Figure 3.10 Flowchart of PSO

Second term in Equation 3.4 represents *cognition*, or the private thinking of the particle when comparing its current position to its own best. Third term of Equation 3.4 represents the social collaboration among the particles, which compares a particle's current position to that of the best particle (Elbeltagi et al. 2005).

The position of particle is updated by the formula:

$$X_{i+1} = X_i + V_i \tag{3.5}$$

Operations mentioned above constitutes one cycle of PSO. Number of cycles is determined by taking the difficult of the problem and analysis duration into account. Flowchart of PSO is given in Figure 3.10.

3.8 Ant Colony Optimization

Marco Dorigo and colleagues introduced the first Any Colony Optimization (ACO) algorithms in the early 1990's which is inferred by the social behavior of ant colonies. Ants live in colonies and their behavior is governed by the goal of colony survival rather than being focused on the survival of individuals. When searching for food, ants randomly search through the possible paths between the food and their nests. Ants leave a chemical called *pheromone* which they can also smell. If an ant should make a decision about selecting a path to follow, it selects the one which contains highest amount of pheromone. The shortest path would have the highest pheromone concentration if all paths are initially preferred by same number of ants. This is because the ants would travel the shortest path in shorter duration and the path would be travelled by more ants when compared with the other paths. As this is the case, the pheromone concentration of the path would be increased by the ants and the probability of the shortest path would be higher than the other paths (Ya-ping and Ying 2006).

During the return trip, the quantity of pheromone that an ant leaves on the ground may depend on the quantity and quality of the food. The pheromone trails will guide other ants to the food source. The indirect communication between ants via pheromone trails, known as *stigmergy*, enables them to find shortest paths between their nest and food sources.

Similar to genetic algorithms and PSO, ACO has population size N, which is generated randomly. Population is initialized by assigning pheromone by generating random numbers. Pheromone values are assigned to the each parameter of ant, which is an S-dimensional space, where S is the number of variables. The probability of a discrete variable being assigned according to its pheromone value is computed by the formula (Dorigo et al. 1996):

$$p_{ijk}(t) = \frac{[\tau_{ijk}(t)]^{\alpha}[\eta_{ijk}(t)]^{\beta}}{\sum_{j \in (1,n_i)} [\tau_{ijk}(t)]^{\alpha}[\eta_{ijk}(t)]^{\beta}} \tag{3.6}$$

where, $p_{ijk}(t)$ is the probability of assigning the k^{th} option to the i^{th} individual's j^{th} parameter at iteration t, $\tau_{ijk}(t)$ is the pheromone value of the i^{th} individual's j^{th} parameter's k^{th} value at iteration t, $\eta_{ijk}(t)$ is a heuristic function, α and β are problem dependent constants, which show the relative importance of pheromone and heuristic function. Heuristic function of the TCT problem with no resource constraint is given as;

$$\eta_{ij}(t) = \frac{\sum_{j} \left|(DirectCost)_{ij} / (Duration)_{ij} - DailyIC(t-1)\right|}{\left|(DirectCost)_{ij} / (Duration)_{ij} - DailyIC(t-1)\right|} \tag{3.7}$$

where *DailyIC* is the daily indirect cost of the project which is computed by taking the overhead, penalty and bonus into account. $DirectCost_{ij}$ is the direct cost of the i^{th} activity's j^{th} crashing option and $Duration_{ij}$ is the duration of the corresponding crashing option.

The crashing options are randomly selected based on their probability of selection computed in the Equation 3.6. Path of each ant is evaluated and the pheromone content is updated by the following equation (Blum 2005a):

$$\Delta\tau_{ijk} = \begin{cases} \dfrac{R}{fitness} & if\ \ option\ is\ selected \\ 0 & if\ option\ is\ not\ selected \end{cases} \qquad (3.8)$$

where, R is a problem dependent constant and *fitness* is the result of the evaluation function of that ant.

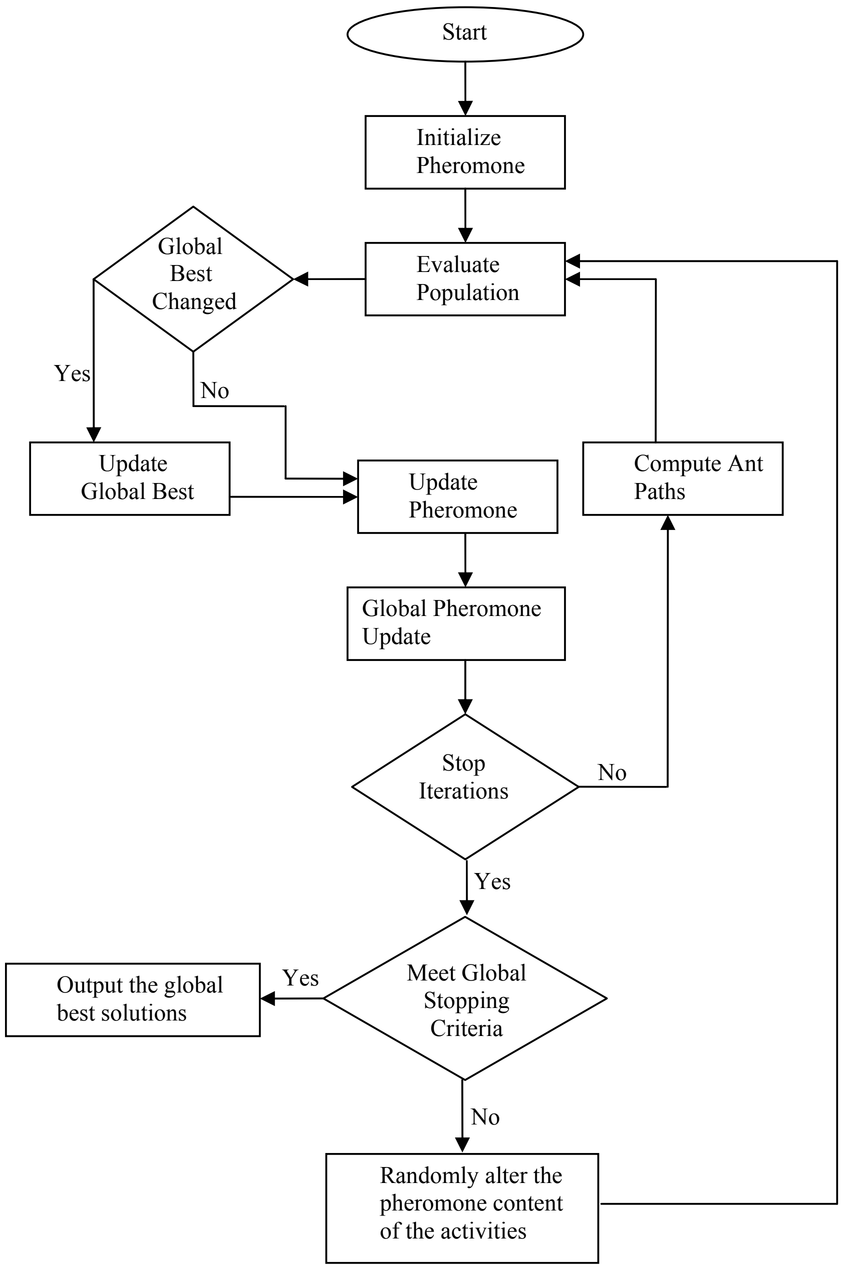

Figure 3.11 Flowchart of ACO

After the evaluation of all ants in the colony the pheromone values are updated by the formula (Dorigo et al. 1996):

$$\tau_{ijk}(t+1) = \rho\tau_{ijk}(t) + \Delta\tau_{ijk} \tag{3.9}$$

where, ρ is the evaporation rate of pheromone which prevent getting stuck into local minima. ρ is chosen in the interval [0,1]. The reason for allowing pheromone evaporation is to avoid too strong influence of the old pheromone to avoid premature solution stagnation. When an entire iteration is over, that is to say after all ants have completed their travels, the pheromone value in options belonging to the best solution in that iteration would be changed according to the following update rule (Sun et al. 2001):

$$\tau_{ijk}(t+1) = (1-z)\tau_{ijk}(t) + z\Delta\tau \tag{3.10}$$

$$\Delta\tau = Rf_{best} \tag{3.11}$$

where z denotes the evaporation rate in the global-updating process; $\Delta\tau$ is equal to the pheromone value changed to the best option of this task in the same iteration; R is the constant representing the pheromone reward factor; and f_{best} is the fitness value of the best ant in the t^{th} iteration.

3.9 Electromagnetic Scatter Search

The fundamental concepts and principles of Electromagnetic Scatter Search (ESS) were first proposed in the 1970s, based on formulations dating back to the 1960s for combining decision rules and problem constraints. In contrast to other evolutionary methods like genetic algorithms, ESS uses strategies for search diversification and intensification that have proved effective in a variety of optimization problems.

ESS includes three basic elements: generation of a population, sized N, which consists of diverse solutions; extraction of high quality and diverse solutions from N and use them to create a reference set R; combine solutions in R to obtain new improved solutions and maintaining and updating R.

Using the diverse solution generator, a set of initial solutions, *N*, is generated. For this purpose, controlled randomization and frequency memory is used. A solution is constructed by randomly selecting an execution mode for each activity. The probability of selecting a mode is inversely proportional to its frequency count. The size of *N* is typically at least 10 times the size of the reference set (Marti et al. 2006). At this stage, solutions are generated without any consideration to their quality with respect to the objective function.

Scatter search does not allow duplications in the reference set, and its combination methods are designed to take advantage of this lack of duplication. Hashing is often used to reduce the computational effort of checking for duplicated solutions. An efficient way of comparing solutions and avoiding duplications can be the following hashing function;

$$Hash(s) = \sum_{i=1}^{m} mk_i i^i \tag{3.12}$$

In which m is the total number of activities in the solution, mk_i is the execution mode assigned to activity i, is computed and used to compare solutions and avoid duplication in *R*. It is empirically determined that two different solutions almost always have different hash values. After the generation of the population electromagnetism mechanism starts.

Electromagnetism mechanism (EM) is a powerful algorithm for global optimization that converges rapidly to optimum. EM can be used as stand-alone approach or as an accompanying algorithm for other methods. The strength of the algorithm lies in the idea of directing the sample points towards local optimizers utilizing an attraction-repulsion mechanism.

In this process, each sample point is considered as a charged particle. Charge of each sample point is initially calculated. The amount of charge relates to the value of the objective function at that point which also determines the magnitude of attraction or repulsion of the point over the sample population. The direction for each point to move in subsequent iterations is then determined. The direction is specified by evaluating a combination force exerted on the point by other points.

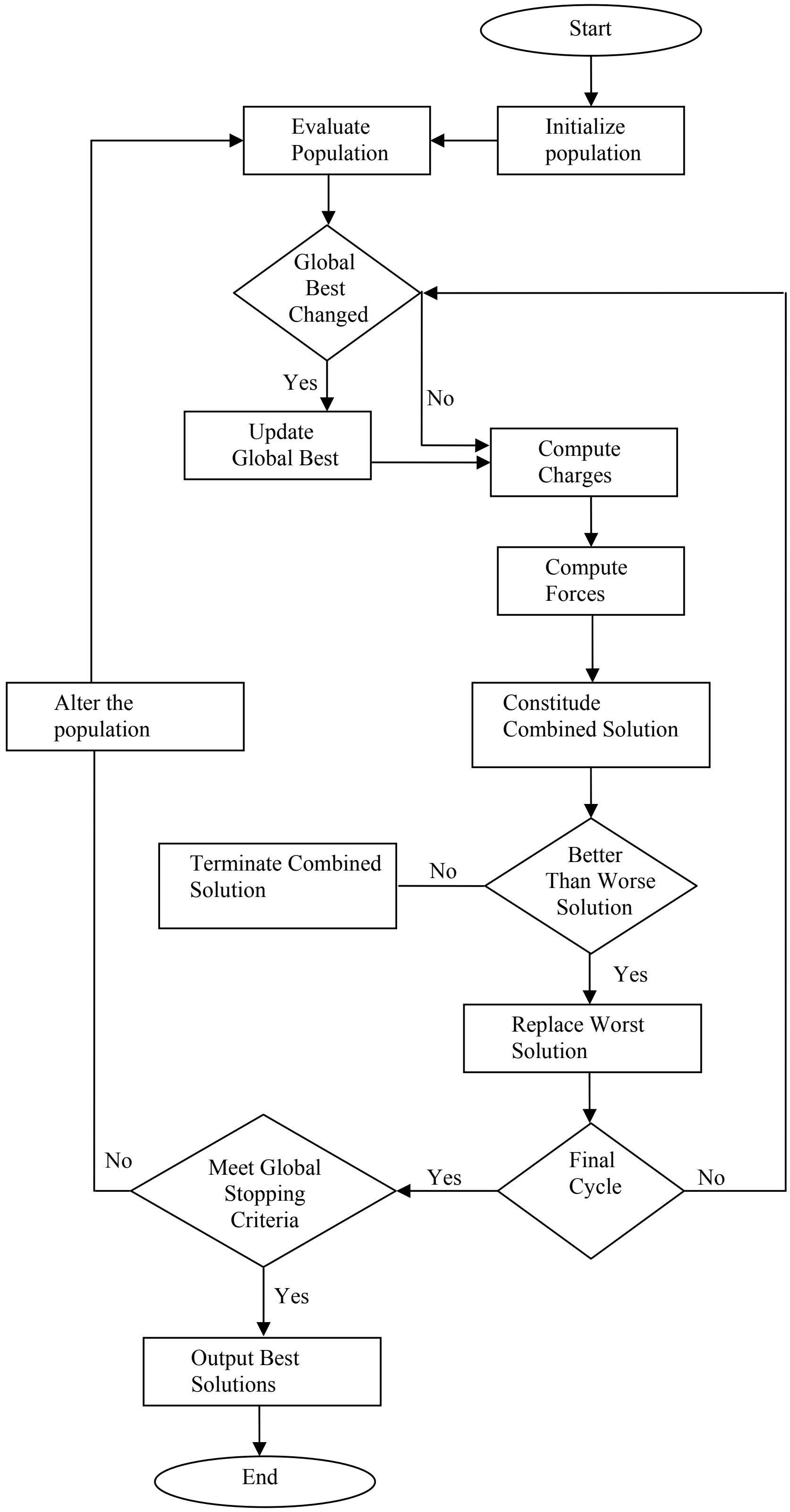

Figure 3.12 Flowchart of ESS

Implementation of EM is done slightly different by Tareghian and Taheri (2007). In their implementation, only one point is considered to act on other points, contrary to the basic EM, where all points in a population exert forces on all other points. For all pairs in R, s_i and s_j, i ≠ j where, $s_i = \left(s_i^1, s_i^2, \ldots s_i^n,\right)$, a force is exerted by point (solution) s_j on s_i either attracting it to its neighborhood or being repulsed by it. According to superposition principle of electromagnetism theory, the force exerted on a point via another point is inversely proportional to the distance between the points and directly proportional to the product of their charges. In the implementation of Taregian and Taheri (2007), the charge of each point is not constant. It changes each time *R* is updated. Instead of using fixed and independent charges for points, a charge, $q_{s_1 s_2} \in [-1,1]$ is defined which depends on the relative efficiency of the objective function value of the corresponding points in the reference set (Tareghian and Taheri, 2007).

$$q_{s_1 s_2} = \frac{f(s_1) - f(s_2)}{f(s_w) - f(s_b)} \tag{3.13}$$

Where $f(s_i)$ is the cost of solution *i*, s_w and s_b are the worst and best solutions in *R* respectively. Attraction of s_1 by s_2 occurs when $f(s_1) > f(s_2)$. Repulsion of s_1 by s_2 occurs when $f(s_1) < f(s_2)$, and no action is taken when $f(s_1) = f(s_2)$. The force exerted by solution s_2 on solution s_1 is calculated as

$$F_{s_1 s_2} = (s_2 - s_1) q_{s_1 s_2} \tag{3.14}$$

According to the value of $F_{s_1 s_2}$, new solutions are created in Euclidian space by moving from s_1 to $s_1 + F_{s_1 s_2}$. The discrete alternative, whose value is closest to the $s_1 + F_{s_1 s_2}$, is assigned to the activity. After the determination of all parameters of the combined solution, s_c, if fitness value of s_c is better then s_w than s_w is replaced with s_c.

ESS involves a double loop which scans all the population. After the evaluation of the double loop, whole population converges to the same solution. This can be either global optima or local optima. In order to escape from local optima, ESS is modified. After executing one set of ESS, one solution is preserved and the others are altered by generating random numbers. Electromagnetic search is repeated several times and with this modification slightly better results are obtained. Flowchart of ESS is given in Figure 3.12.

CHAPTER 4

ANALYSIS OF TCT PROBLEMS

In this chapter, analysis of TCT problems obtained from literature is explained. Test problems are illustrated and the performances of the meta-heuristic models are compared and their results are discussed. In addition to this, representations of the meta-heuristic algorithms are illustrated.

4.1 Abstraction of TCT problems by meta-heuristic algorithms

Nature of the TCT problem requires the determination of the crashing alternatives of the activities which gives the least total project cost. In a TCT type problem, the number of parameters becomes the number of activities of the project which have more than one construction mode alternative. As this is the case, meta-heuristic alternatives should be able to represent the crashing alternatives efficiently. There are many possibilities for the representation of crashing alternatives. In order to implement a method which is computationally feasible and allowing diversification, the domain of the crashing alternatives are divided into discrete portions for the representation of crashing alternatives.

The reason for the division of discrete ranges is that, crashing alternatives are represented discretely because they are more realistic than continuous crashing functions weather linear or nonlinear. In real construction projects, there can be a few alternatives for the selection of equipment such as excavator, crane or concrete plant. Similarly, number of labors can only be assigned as the integer multiplies of the crew sizes for most of the construction activities such as constructional steel erection, welding, or electrical and mechanical works. For this reason, it is not practical and realistic to represent the crashing alternatives of an activity by a continuous function.

Within this context, some of the test problems containing linear crashing options are converted into discrete crashing options without eliminating any crashing alternative.

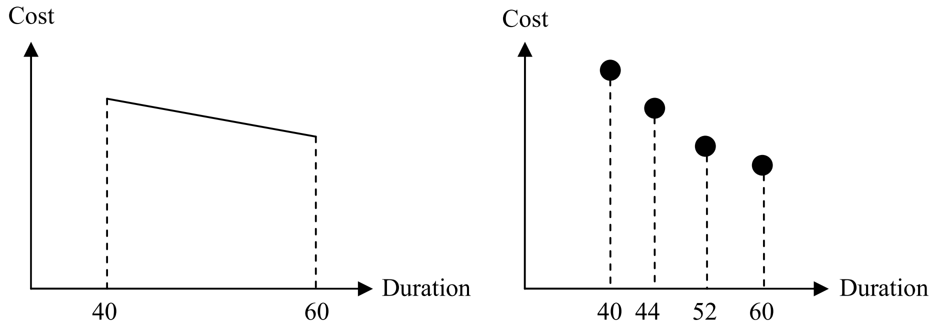

Figure 4.1 Continuous and discrete function

To give an example, if normal duration of a trench excavation activity is 60 days which can be crashed up to 40 days would have 21 different duration-cost alternatives if a continuous function is assigned for the representation of the crashing alternatives. However, in the market it is not possible to procure or hire equipments capable of giving outputs for 21 different completion alternatives. On the other hand, discrete representation shown in Figure 4.1 is more realistic than continuous functions in terms of taking crashing alternatives into account.

For the gene representation of GA, binary representation is preferred although it has some difficulties for the evaluation of discrete crashing alternatives different than 2^n. In order to handle activities with crashing alternatives different than 2^n i.e. 3 or 5; some of the crashing possibilities must be left empty if two bits and three bits representation is assigned respectively. After the mutation or crossover operations, genes carrying these blank portions might have occurred. If this is the case; by generating a random number, the crashing alternative can be determined. However, if crashing alternative would be determined by a random number which by-passes the genetic representation, implementing genetic algorithm would be nonsense.

In order to handle any number of crashing alternatives special method is implemented for the representation of crashing alternatives. A pseudo-continuous interval is generated by assigning high gene lengths for the representation of crashing alternatives. For example, if the genes length is determined as 8 bits it would correspond to 256 intervals. These intervals are classified into classes in which the number of classes are same with the number of crashing alternatives of the corresponding activity. The bit length is determined in a way to increase the number of intervals adequate enough to divide by the number of crashing alternatives as fair as possible. The maximum error in this case is less than the remainder of the division of number of intervals with the number of crashing alternatives divided by the number of crashing alternatives. In Figure 4.2 the intervals for the representation of 3 and 5 crashing alternatives respectively.

Similar methodology is implemented for the representation of crashing alternatives by the PSO and ACO. *Position* and the *pheromone* content are the parameters necessary for the determination of the crashing alternative by the PSO and ACO algorithms respectively. Range, which is the boundary of position and pheromone content, can also be divided into discrete intervals. Number of discrete intervals is assigned as same amount with the GA based methods for the simplicity of the comparison of methods.

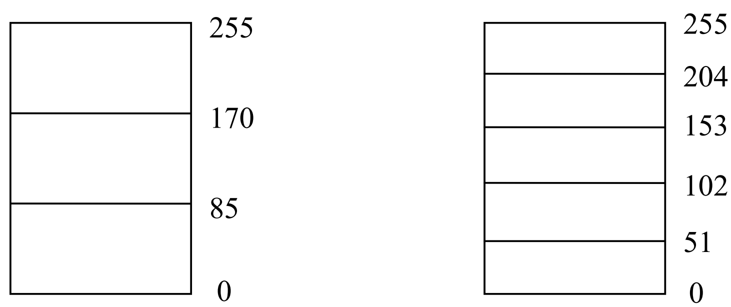

Figure 4.2 Illustration of borders of the crashing alternatives

Probabilities assigned for each crashing alternatives would be almost equal and the difference between the probabilities can easily be ignored. The difference can be further decreased by increasing the bit size of the genes which increases the storage and computational demand. Therefore, there is a trade-off between the enlarging the gene size and the computational demand. In order to keep the gene size within reasonable length, at most 8 bit per activity is preferred. For 5 crashing alternatives this makes 51 units of interval for 4 crashing alternatives and 52 units for the last crashing alternative. For 3 crashing alternatives, 2 of them will have 85 units of interval and the remaining one will have 86 units of crashing alternatives. The examples show that, the probabilities of the selection of crashing alternatives can be considered as equal.

4.2 TCT Analysis with 7-Activity Project

First analysis for the TCT type problems is conducted with a 7-activity project developed by Burns et al. (1996). The network contains 7 activities with logical relationships of only FS with no lag which is shown in **Figure 4.3**. The crashing options for the activities are given in a hybrid way, such that both linear and discrete crashing alternatives can be assigned to an activity.

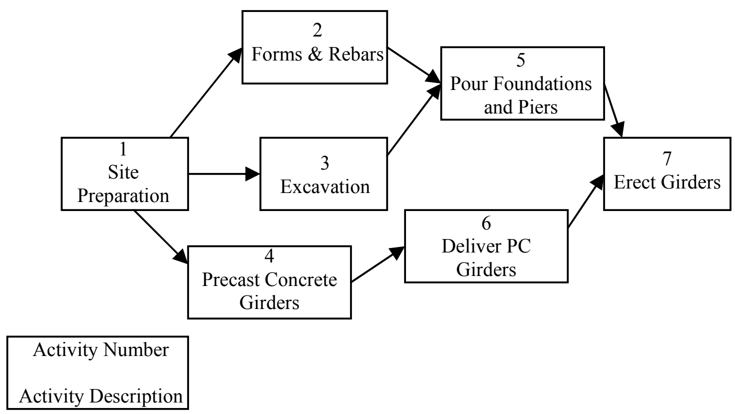

Figure 4.3 Network of 7-activity project

The crashing alternatives of the network are given in Table 4-1;

Only second activity has linear crashing options and the total crashing alternatives of that activity is 8. The TCT problem is relatively an easy problem to solve as the total combinations is 3*8*3*3*4*3*3 = 7776 which means if an exhaustive enumeration would be performed for the exact solution, only 7776 trials is necessary. Burns et al. (1996) assumes indirect cost of $1000/day as overhead cost and obtains two different optimum solutions which are 78 days of construction duration with $107500 direct cost and $185500 total cost; 84 days of construction duration with $101500 direct cost and $185500 total cost.

The aim of the trial is not only obtain the exact solution but also obtain the solution in less than 7776 trials. GA, GASA, HGAQSA, GMASA, GASAVNS, ACO, EMS and PSO are implemented for the solution of the network.

Table 4-1 Activity options for the 7-activity network

Activity Description	Activity Number	Option	Duration	Cost	Slope Relationship
Site preparation	1	Crew 1 + Eq 1	14	23000	D
	1	Crew 2 + Eq 2	20	18000	D
	1	Crew 2 + Eq 2	24	12000	D
Forms and Rebar	2	Method 1	15	3000	C
	2	Method 2	18	2400	
	2	Method 3	20	1800	D
	2	Method 4	23	1500	C
	2	Method 5	25	1000	D
Excavation	3	Equipment 1	15	4500	D
	3	Equipment 2	22	4000	D
	3	Equipment 3	33	3200	D
Precast Concrete Girder	4	Method 1	12	45000	D
	4	Method 2	16	35000	D
	4	Method 3	20	30000	D
Pour Foundations and Piers	5	Method 1	22	20000	D
	5	Method 2	24	17500	D
	5	Method 3	28	15000	D
	5	Method 4	30	10000	D
Deliver PC Girders	6	Railroad	14	40000	D
	6	Truck	18	32000	D
	6	Barge	24	18000	D
Erect Girders	7	Crane 1 + Crew 1	9	30000	D
	7	Crane 2 + Crew 2	15	24000	D
	7	Crane 3 + Crew 3	18	22000	D

4.2.1 Solution by GA

Population size is taken as 25 which is close to the gene length computed by the multiplication of parameter number by bit per activity. Genetic iteration is determined to be high enough that total evaluation number will reach 10000 evaluations. The parameters assigned for the GA is given in Table 4-2.

Table 4-2 Parameters of GA for the 7 Activity network

Genetic Iteration	1000
Population Size	25
bitPerActivity	4
Crossover	0,6
Mutation	0,07

The analysis is repeated 10 times in order to measure possible deviations of the solution. The results of the solutions obtained by the GA are shown in Table 4-3. Although the 7-activity TCT problem was relatively an easy problem and the total project iteration is more than the iteration of exhaustive evaluation, GA can find the optimum solution at only 4 trials of 10. At the end of 1000 project evaluation, optimum solution could not be achieved in any trials. Only one optimum solution is obtained after the 2500 and 5000 iterations. After the 10000th project evaluation, only 4 of the trials reach the optimum solution. One analysis took around 0,3 seconds on one processor of 2,4 GHz Intel Core2 Duo CPU. Although the 7-activity project is relatively an easy TCT problem, GA could not present successful results.

Table 4-3 Solution obtained by GA for 7-activity project

Analysis No	**1000**	**2500**	**5000**	**10000**
1	187100	185500	185500	185500
2	193000	187300	186500	185500
3	192300	190300	189300	186500
4	193400	193400	191500	190300
5	188900	187300	186500	186500
6	189300	189300	187300	187300
7	186500	186500	186500	185500
8	188500	188500	185500	185500
9	190100	188900	187100	186500
10	190100	190100	190100	186300

4.2.2 Solution by GASA

Same problem is analyzed by GASA. The project is again solved by 10 times in order to monitor the deviations of the obtained results. The parameters shown in Table 4-4 are assigned.

Table 4-4 Parameters of GASA for the 7 Activity network

Genetic Iteration	500
Population Size	25
bitPerActivity	4
Crossover	0,6
Mutation	0,8
Boltzmann Constant	0,9

Mutation rate is higher than the GA because the acceptances of the mutations are controlled by the Simulated Annealing process. SA prevents deterioration of high quality genes by harmful mutations. To increase the local search ability of the algorithm, mutation rate is increased by depending on the protection of SA. In order to increase the cooling speed and help the algorithm converge faster, Boltzmann Constant can be assigned less than 1. The 7-activity project is a small problem, as a result of this number of local minima are relatively less than the ones in complex projects. Probability of getting stuck into local minima is low. To decrease the probability of acceptance of a detrimental mutation, BC is assigned as 0.9. Results shown in Table 4-5 are obtained by GASA.

Table 4-5 Solution obtained by GASA for 7-activity project

Analysis No	**1000**	**2500**	**5000**	**10000**
1	188900	186300	185500	185500
2	185500	185500	185500	185500
3	187300	185500	185500	185500
4	185500	185500	185500	185500
5	185500	185500	185500	185500
6	186500	185500	185500	185500
7	189100	185500	185500	185500
8	186500	186300	185500	185500
9	186500	186300	185500	185500
10	187300	186300	185500	185500

Successful results are obtained by GASA at the end of 10 trials. The optimum solution is obtained in all trials. At the end of the 1000^{th} iteration optimum solution is obtained in 3 of the 10 trials and 6 of the 10 trials at the end of 2500^{th} iteration. At the end of 5000th iteration, optimum solution is obtained in all trials. One analysis took around 0,3 seconds on one processor of 2,4 GHz Intel Core2 Duo CPU.

4.2.3 Solution by HGAQSA

HGAQSA differs from GASA by its enhanced local search capability. Because of its hybrid local search algorithm, number of genetic iteration which is the repletion of whole GA cycle is significantly less than GA and GASA. As a result of this, cooling of HGAQSA must be done in less cycles compared with GASA. In order to increase cooling speed, BC is assigned as 0.8 which is less than the BC assigned to GASA. Parameters assigned for the HGAQSA are shown in Table 4-6. Genetic algorithm is stopped after *SAperiod* of generation evaluation. At this point, random mutations are applied to the *RWalkNo* individuals and the number of mutations per individuals is *SAiteration*. The individuals are selected randomly but, the best individual is guaranteed to be selected at each QSA iteration. Following results obtained by HGAQSA are shown in Table 4-7.

Table 4-6 Parameters of HGAQSA for the 7 Activity network

Genetic Iteration	100
Population Size	25
bitPerActivity	4
Crossover	0,5
Mutation	0,8
Boltzmann Constant	0,8
SAiteration	14
SAperiod	10
RWalkNo	15

Table 4-7 Solution obtained by HGAQSA for 7-activity project

Analysis No	**1000**	**2500**	**5000**	**10000**
1	187500	185500	185500	185500
2	185500	185500	185500	185500
3	185500	185500	185500	185500
4	185500	185500	185500	185500
5	185500	185500	185500	185500
6	185500	185500	185500	185500
7	185500	185500	185500	185500
8	185500	185500	185500	185500
9	185500	185500	185500	185500
10	185500	185500	185500	185500

HGAQSA obtained the optimum solution at 9 of the 10 trials at the end of 1000^{th} iteration and at all trials after the 2500^{th} iteration. Similar to the previous analysis, computational duration is measured as 0,3 seconds.

4.2.4 Solution by GMASA

GMASA is a similar algorithm to HGAQSA in which it differs only in the local search operator that sequentially searches through the search space. On the other hand, HGAQSA performs a random walk on the parameters. The search algorithm of GMASA significantly increases the evaluation number. However, it increases the probability of convergence into global optima. Parameters assigned for the GMASA are shown in Table 4-8.

GMASAiteration is the number of memetic searches for each individual. The algorithm seeks for better solutions by mutating the genes in a sequential way. This procedure is repeated 2 times for the selected individual making 14 mutations for each MA search.

Table 4-8 Parameters of GMASA for the 7 Activity network

GeneticIteration	90
Population	25
bitPerActivity	4
GMASAiteration	14
Boltzmann	0,7
Crossover	0,5
Mutation	0,8
Maperiod	10

When the Table 4-9 is examined, it is seen that the convergence of GMASA is relatively slow. Although GMASA obtained optimum solution at the end of all analysis, the method could not be considered as successful. Duration of one analysis is measured as 0.3 second.

Table 4-9 Solution obtained by GMASA for 7-activity project

Analysis No	**1000**	**2500**	**5000**	**10000**
1	187900	187300	187300	185500
2	192000	187500	187500	185500
3	186500	186500	185500	185500
4	187500	187300	185500	185500
5	190500	187900	185500	185500
6	187100	186500	186500	185500
7	191300	185500	185500	185500
8	187100	186300	186300	185500
9	187900	187900	187500	185500
10	185500	185500	185500	185500

4.2.5 Solution by GASAVNS

The 7-activity project is analyzes by GASAVNS and the model parameters are given in Table 4-10. The variable neighborhood search algorithm performs at most 3 mutation at the same time which is determined by the *MaxNeighbor* parameter. The local search is repeated 3 times which is determined by *SAiteration* parameter. Similar to GMASA, local search is applied to all individuals of the population.

Table 4-10 Parameters of GASAVNS for the 7 Activity network

GeneticIteration	250
Population Size	25
bit/activity	4
SAiteration	3
crossover	0,4
mutation	0,8
VNSperiod	5
Boltzman	0,5
MaxNeighbor	3

The genetic algorithm is stopped after evaluation of *VNSperiod* generations and VNS local search begins. In this example, after 5 GASA cycles, VNS local search is applied. The analysis results are shown in Table 4-11.

Table 4-11 Solution obtained by GASAVNS for 7-activity project

Analysis No	**1000**	**2500**	**5000**	**10000**
1	186500	185500	185500	185500
2	185500	185500	185500	185500
3	187100	185500	185500	185500
4	187100	185500	185500	185500
5	186300	185500	185500	185500
6	185500	185500	185500	185500
7	185500	185500	185500	185500
8	185500	185500	185500	185500
9	186500	185500	185500	185500
10	185500	185500	185500	185500

GASAVNS obtains the optimum at all trials after 2500^{th} evaluation. In addition to this, after the 1000^{th} iteration in five of the ten trials optimum schedule is obtained. The results show that GASAVNS converges to optimum fast.

4.2.6 Solution by ACO

Ant colony optimization seeks the global optimum by assigning more probability of selection to the crashing alternatives which ends up with schedules of lower total project costs. Probability values are determined by considering the pheromone content of the alternatives. Upper bound for the pheromone content is limited by the *MaxPheromone* parameter. The pheromone amount added for the best solution's crashing alternatives is equal to *pheromoneConstant*. The other solution's added pheromone amounts are determined by taking the ratio of individual's total project cost and the current best project total project cost into account.

Alfa and Beta are the parameters for the computation of the probability of selection of the crashing options by considering the pheromone amounts. In order to prevent getting stuck into local minima, pheromone amounts are evaporated by taking the Evaporation parameter into account. After one cycle of ACO, 30% of the pheromone is evaporated. Parameters assigned for the ACO algorithm is shown in Table 4-12.

Table 4-12 Parameters of ACO for the 7 Activity network

Iteration	500
Population	25
Max_Pheromone	64
q0	0,2
Alfa	1
Beta	0,001
pheromoneConstant	12
Evaporation	0,7
Best_Evap	0,97

Analysis results show that ACO converges to optimum significantly fast that after the 1000th evaluation the algorithm had obtained 9 optimum solutions out of 10 trials. In addition to this, optimum solution is obtained when the stopping criteria is reached.The analysis result of ACO is given in Table 4-13.

Table 4-13 Solution obtained by ACO for 7-activity project

Analysis No	**1000**	**2500**	**5000**	**10000**
1	194000	194000	191500	190500
2	185500	185500	185500	185500
3	185500	185500	185500	185500
4	185500	185500	185500	185500
5	185500	185500	185500	185500
6	185500	185500	185500	185500
7	185500	185500	185500	185500
8	185500	185500	185500	185500
9	185500	185500	185500	185500
10	185500	185500	185500	185500

4.2.7 Solution by ESS

Model parameters of ESS consists of only population size and the repetition number of ESS by randomly alteration of the results obtained by ESS. Population size for the ESS is chosen as 14 and the scatter search is repeated 7 times. After the 10 trials, obtained results are shown in Table 4-14.

Table 4-14 Solution obtained by EMS for 7-activity project

Analysis No	**1000**	**2500**	**5000**	**10000**
1	195500	186500	186300	185500
2	188100	187500	186500	185500
3	191500	186500	185500	185500
4	190300	188500	185500	185500
5	193000	187300	185500	185500
6	191900	190800	190300	185500
7	191500	187100	185500	185500
8	201700	192000	187500	185500
9	203500	192800	185500	185500
10	195750	190300	188500	185500

ESS obtained optimum when the stopping criterion is met; however the convergence of ESS is very slow. Only in five trials out of ten, optimum solution is obtained after the 5000th evaluation. In addition to this, ESS gives the worst results at the end of the 1000th evaluation, which means the initial convergence capability of the algorithm is the worst.

4.2.8 Solution by PSO

Particle swarm optimization seeks the global optimum by directing each particle's position, crashing alternatives, towards the global best and that individual's current best. Each particle's velocity is determined by considering the distance between the global best and its overall best.

In order to prevent missing out any crashing alternatives which is on the search direction, the maximum velocity is limited by V_{max} parameter. The maximum distance between the crashing alternatives is determined by the parameter *Resolution*. Parameters c_1 and c_2 are the importance factors of the distance between global optimum and current best respectively. Parameters of the PSO are given in Table 4-15.

Table 4-15 Parameters of PSO for the 7 Activity network

Iteration	100
Population Size	25
Random Iteration	5
Resolution	64
c1	1,9
c2	1,9
Vmax	16

PSO is repeated "Random Iteration" times by randomly changing the obtained positions in order to prevent getting stuck into local minimum. The obtained results are shown in Table 4-16.

After the 5000th evaluation PSO obtained the optimum solution in all trials. Only in one trial out of ten, global optimum could not be obtained at the end of the 2500th evaluation.

Table 4-16 Solution obtained by PSO for 7-activity project

Analysis No	1000	2500	5000	10000
1	187100	187100	185500	185500
2	185500	185500	185500	185500
3	185500	185500	185500	185500
4	185500	185500	185500	185500
5	185500	185500	185500	185500
6	185500	185500	185500	185500
7	186300	185500	185500	185500
8	186300	185500	185500	185500
9	186300	185500	185500	185500
10	186500	185500	185500	185500

4.2.9 Conclusion

7-activity project was relatively an easy TCT problem. For this reason, obtaining global optima does not necessarily mean that the meta-heuristic algorithm is successful. When the stopping criteria is met which is more than the evaluation number of exhaustive evaluation, all meta-heuristic algorithms obtained global optimum in some or all of the trials. For this test problem, performance of HGAQSA and GASAVNS are the best and they showed pleasing results in terms of convergence speed and precision of the results.

PSO is also a fast converging algorithm. Its convergence to near-optimum is significantly fast but it takes much iteration for PSO to converge into global optimum. For this reason, PSO is the third algorithm after HGAQSA and GASAVNS.

ACO has the best initial convergence speed. At the end of the 1000th evaluation, results of ACO are better than the any other algorithms. However, precision of ACO is not satisfactory. Improvement of the initially obtained near-optimum results are significantly slow at later iterations which causes the risk of getting stuck into local optima.

GASA obtained satisfactory results that after the 5000^{th} evaluation, it obtained global optimum in all trials. This property of GASA makes it a dependable optimization method in contrast to its slow convergence speed for this problem. Finally, GMASA, ESS and GA did not present satisfactory results in this analysis.

To conclude, 7-activity project is not a proper problem to test the convergence abilities of the meta-heuristic algorithms. However, the important criteria in this case were to reach global optimum in the project evaluation number which is less than the number of combinations of the project. HGAQSA and GASAVNS are successful meta-heuristic algorithms with their precise and fast convergence into global optima.

4.3 TCT Analysis with 18 Activity Project

Second TCT type problem examined is an 18-activity Project which was generated by Burns et al. (1996) and altered by Hegazy (1999). There are two to five different cost and duration alternatives for each activity. 18-activity project's logical relationships and crashing options of the activities are shown in Table 4-17.

Total number of possible different cost-duration alternatives is;
$5^5 * 4^2 * 3^{10} * 2^1$ = 5,9 Billion alternatives.

Complete evaluation of this much high value takes significantly long time even if it was evaluated on a high speed computer. For this reason, implementing an exhaustive enumeration is almost impossible. The 18-activity project is analyzed for 2 cases:

- Minimum total project cost for constant $200/day overhead cost and $1000/day bonus for earlier finish from 110 days and $20000/day liquidated damages for later finishes from 110 days.
- Minimum total project cost for constant $200/day overhead cost.

Table 4-17 18 Activity project

Act. No.	Predecessor	Alternative 1		Alternative 2		Alternative 3		Alternative 4		Alternative 5	
		Dur. (days)	Cost ($)	Dur. (days)	Cost ($)	Dur. (days)	Cost ($)	Dur. (days)	Cost ($)	Dur. (days)	Cost ($)
1	–	14	2,400	15	2,150	16	1,900	21	1,500	24	1,200
2	–	15	3,000	18	2,400	20	1,800	23	1,500	25	1,000
3	–	15	4,500	22	4,000	33	3,200	–	–	–	–
4	–	12	45,000	16	35,000	20	30,000	–	–	–	–
5	1	22	20,000	24	17,500	28	15,000	30	10,000	–	–
6	1	14	40,000	18	32,000	24	18,000	–	–	–	–
7	5	9	30,000	15	24,000	18	22,000	–	–	–	–
8	6	14	220	15	215	16	200	21	208	24	120
9	6	15	300	18	240	20	180	23	150	25	100
10	2, 6	15	450	22	400	33	320	–	–	–	–
11	7, 8	12	450	16	350	20	300	–	–	–	–
12	5, 9,10	22	2,000	24	1,750	28	1,500	30	1,000	–	–
13	3	14	4,000	18	3,200	24	1,800	–	–	–	–
14	4, 10	9	3,000	15	2,400	18	2,200	–	–	–	–
15	12	12	4,500	16	3,500	–	–	–	–	–	–
16	13, 14	20	3,000	22	2,000	24	1,750	28	1,500	30	1,000
17	11, 14, 15	14	4,000	18	3,200	24	1,800	–	–	–	–
18	16, 17	9	3,000	15	2,400	18	2,200	–	–	–	–

The two contract types are analyzed by the eight meta-heuristic algorithms. The GA based methods: GA, GASA, HGASA, HGAQSA, GMASA, GASAVNS have the same gene representation. Binary coding is preferred and each activity is represented by 6 bits. As a result the total gene length is 18*6 = 108 bits.

The number of chromosomes, namely crossover points, is a critical issue to determine. In order to see the effect of number of chromosomes, different number of chromosomes from 1 to 4 is tested. The gene is divided into equal intervals at each trial and the crossover point for each chromosome is determined by generating random numbers. Each trial is repeated 10 times and the following results shown in Table 4-18 are obtained.

Table 4-18 Effect of chromosome number

Analiz No	**1CH GA**	**2CH GA**	**3CHGA**	**4CH GA**
1	138865	139070	142670	142520
2	147170	130670	143558	140550
3	149070	147590	140170	148005
4	138870	144850	144300	139858
5	130520	140500	148050	137170
6	144370	144870	145790	144650
7	146815	149908	145770	143615
8	142960	148320	139370	146608
9	141908	146015	151338	141570
10	148065	142958	141308	144100
Mean	142861,3	143475,1	144232,4	142864,6
St Dev	5649,25	5623,06	3673,52	3237,52

From the analyses, it is seen that for the genetic algorithm, chromosome number does not have a significant affect on the mean value of the solutions. However, the standard deviation of the results decreases as the number of chromosome increases. Precision of the results have significant importance, because in this thesis it is aimed to develop a meta-heuristic algorithm which converges to global optimum or near-optimum at every trial. As a result, if the results have high precision this reflects that the algorithm has tendency to converge to a stable solution. Therefore, four-chromosome genetic representation is preferred during the thesis study.

18-activity network is analyzed by the meta-heuristic methods and the results are briefly introduced. The 6 bit per activity is assigned for the GA based methods. Pheromone interval for the ACO and position interval for the PSO is also assigned as 64.

The two contract conditions are re-solved by adding the project on to itself 5 and 19 times respectively. As a result of this, 108 and 360-Activity projects are obtained. Problem solution capabilities of meta-heuristic algorithms are tested on large projects also.

For the first case there is not any penalty function for the restrictions, cost function is adequate for the total evaluation of the contract condition.

$$CF = \begin{cases} DC + 20000 * (PD - 110) + PD * 200 \quad if \quad PD > 110 \\ DC - 1000 * (110 - PD) + PD * 200 \quad if \quad PD < 110 \\ DC + PD * 200 \quad if \quad PD = 110 \end{cases} \tag{4.1}$$

where, *PD* is the project duration. Optimum solution of this case was 110 days of project duration and $106270 direct cost and $22000 indirect cost which is the same solution with the previous case. Contract conditions of the first case restrict only delayed projects and awards the projects finished earlier than 110 days.

For the second case, there is only constant overhead cost of $200/day. There is not any limitation on the project duration by the contract clauses. Aim is to obtain the optimum project scheduling which gives the least total project cost. Cost function of the second case is shown as;

$$CF = \sum DC + 200 * PD \tag{4.2}$$

Solution of this problem was 126 days of project duration with $102570 direct cost and $25200 indirect cost which makes the total cost $127770.

4.3.1 Analyses by GA

GA is a very simple method to implement as there are very few parameters to assign. GA does not demand a significant meta-heuristic knowledge to run the analysis for a project manager. There are not complicated parameters to decide the values of them to improve analysis results.

Analysis results are given in Table 4-19 for the cases 1 and 2. It can easily be inferred from the analysis results that GA can not give satisfactory results. Although 18-Activity project is not a challenging problem, the best solution is not close to optimum solution.

At first glance Case 1 seems to be an easy problem to solve, since the project completion dates later than 110 days are penalized heavily. The penalty function restricts the solution space and only early finishes than 110 days have high probability of survival. This effect is seen especially in the 18-Activity projects which have earlier than 110 days solution. However, those solutions are far from optimum solutions.

Best solution deviates about 13,4% and the mean deviates about 19,5% from the optimum solution. This represents that the GA can not converge into global optimum successfully and the results can deviate significantly. However, project duration has not deviated as much as the project cost. This means that GA is not successful at extending the activity durations of activities which have slack time. As a result of this, global optimum could not be obtained.

GA could not give successful results when the project is duplicated six times. This result has already been expected when the 18-Activity project results are examined. However, the results are surprising that none of the analysis could not finish the project earlier than 660 days and are penalized by $20000 per delay day. As a result of this the results are extremely bad that even the best solution costs 2,85 times of the optimum solution.

Table 4-19 Analysis results by GA

		Case 1		Case 2	
	Project Size	**Cost**	**Duration**	**Cost**	**Duration**
Best Solution	18 Activity	145400	106	129116	122
	108 Activity	2192400	722	798290	989
	360 Activity	10062232	2541	2668600	3341
Mean of 10 Analysis	18 Activity	153250	106,5	129833	125
	108 Activity	2416666	732	816440	1001
	360 Activity	10395559	2557	2698744	3361

Table 4-20 Model Parameters of the GA

Genetic Iteration	1000
Population Size	100
bitPerActivity	6
Crossover	0,6
Mutation	0,07

It is obvious that GA is trapped in a local optimum and could not obtain a better solution although there were significantly better solutions. The main reason of this is that GA can not systematically improve the solutions and mutations may be harmful for complex projects if an improvement is obtained. Probability of a beneficial mutation decreases significantly as the gene quality of the individuals improve compared with their initial, in other words their randomly generated, state.

It is clear that in order to finish the project earlier than 660 and avoid paying liquidated damages, successive beneficial mutations are mandatory. However, probability of a beneficial mutation decreases as the gene quality improves. Unfortunately, the improvement is terminated by a harmful mutation and the gene quality again decreases. However, it is surprising that there are not any infeasible early finishes which was observed in the 18-Activity project.

The results are even worse at the 360-Activity project. This can be explained by the same reasoning with the previous analysis. In addition to this, in this case the problem is more complicated and the gene structure is more difficult to improve. As a result the analysis results are not satisfactory.

Although the results are not satisfactory, the deviation of the 10 analysis is not much. Difference between the best solution and the mean of the results are not very much. This is an unexpected situation, because it is expected that the solutions of the GA deviate in a wide range as the solutions are not satisfactory. In contrast with this, the solutions converged in a similar local optimum in which project cost and project duration do not deviate much.

Reason of this can be explained as the four point crossover operation, which keeps the mean of the solutions in a steady state and decreases the variation of the results. Main difference between the Case 1 and Case 2 is that there is not an apparent feasible region in Case 2. As a result of this the feasible solution space is significantly wider in Case 2. There would not be abrupt changes in the project cost with the fluctuations of the project duration.

For the 18-Activity project GA gives satisfactory results. The Best solution among 10 trials is very close to the optimum solution. Project cost is 1% higher and the project duration is 4 days shorter than the optimum solution. When these results are compared with the previous case's results, in this case GA can be said to give satisfactory results. In addition to this, there is not a significant deviation in the mean of the analyses. Especially mean of the project cost did not deviate much and the mean project duration is surprisingly close to the optimum solution.

Similarly 108 and 360-Activity project solutions are not deviated as much as the previous case's results. However, the results are far from being acceptable. Deviation of the best result of the 108-Activity and 360-Activity projects are a little less than 5% and the deviation of mean is a little more than 5%. However, project duration is not close to the optimum solution. It is obvious that by extending the activity durations on the critical path project duration, thus the indirect cost is increased and the direct cost is decreased. It is probable that the GA got stuck into local minima and could not achieve better solution. Consequently, when the analysis duration and the analysis results are compared it can be said that the results are not satisfactory.

Analysis duration of GA is measured as 40 seconds for the 18-Activity project, 12 minutes for 108-Activity project and 37 minutes for 360-Activity project.

4.3.2 Analyses by GASA

Due to SA characteristics, analysis with GASA had ended up with successful results which are given in **Table 4-21**. GASA obtained optimum solutions at every trial for the 18-Activity and 108-Activity projects. In addition to this, GASA obtained very close results to the global optimum for the 360-Activity projects. If the population size and the iteration is increased it could be possible for GASA to obtain the global optimum of 360-Activity project also.

Analysis results of GASA deviated only 0,5% from the global optimum in terms of total project cost for the 360-activity project. In the first case GA is penalized by the liquidated damage because of the late project completion. However, in none of the analysis GASA has late finish penalty. GASA end up with slightly shorter project durations compared with the project duration of global optimum. Mean of the 360-

Activity project solutions are very close to each other which means that the success of GASA is not by chance. GASA had obtained global optimum at every trial of 18-Activity and 108-Activity projects and very close results to global optimum at 360-Activity projects.

Table 4-21 Analysis results by GASA

	Project Size	Case 1		Case 2	
		Cost	**Duration**	**Cost**	**Duration**
Best Solution	18 Activity	128270	110	127770	126
	108 Activity	769620	660	766620	756
	360 Activity	2686452	2192	2558526	2522
Mean of 10 Analysis	18 Activity	128270	110	127770	126
	108 Activity	769620	660	766620	756
	360 Activity	2687957	2192	2558925	2530

Model parameters of GASA are determined according to the project size. The total project evaluation number is proportional with the square of the project size. The evaluation duration of GASA analysis is proportional with the cubical power of project size. As a result exact solution of very large projects may not be suitable by GASA. The model parameters of GASA are given in Table 4-22.

Table 4-22 Model Parameters of the GASA

Genetic Iteration	1000
Population Size	150
bitPerActivity	6
Crossover	0,6
Mutation	0,8
Boltzmann Constant	0,9

Similar to previous case, GASA end up with successful results in Case 2 too. For the 18-Activity and 108-Activity projects GASA obtained the global optimum for every trial. This showed the ability of GASA on the solution of medium sized TCT problems. In addition to this, very close results to the global optimum are obtained at the analysis on 360-Activity projects. The deviation of total project cost is less than %0,1 both for the best solution and the mean of the 10 trials. The error amount is significantly small and can be accepted as near-optimum solution.

Best solution has a schedule two days later project completion time compared with the global optimum. However, mean project completion time is 10 days later than the optimum solution. This indicates that there is a slight deviation in the solutions of GASA in terms of project duration; however 10 days is not a significant time when the 2520 days of optimum project duration is considered.

GASA can be considered to be a successful method in that; it can obtain global optimum for the small and medium-sized projects and obtained satisfactory results for the large-sized projects. However, better results must be achieved for a proper project planning.

Analysis duration of GASA is measured as 8 seconds for the 18-Activity project, 7 minutes and 30 seconds for 108-Activity project and 43 minutes for the 360-Activity project.

4.3.3 Analyses by HGAQSA

HGAQSA end up with very successful results which can be seen from Table 4-23. Optimum solution is achieved at all trials including the 18-Activity, 108-Activity and 360-Activity projects. In addition to its successful results, HGAQSA has high convergence speed to the global optimum.

Table 4-23 Analysis results by HGAQSA

		Case 1		**Case 2**	
		Cost	**Duration**	**Cost**	**Duration**
Best Solution	18 Activity	128270	110	127770	126
	108 Activity	769620	660	766620	756
	360 Activity	2565400	2200	2555400	2520
Mean of 10 Analysis	18 Activity	128270	110	127770	126
	108 Activity	769620	660	766620	756
	360 Activity	2565400	2200	2555400	2520

Random search of HGAQSA was the only difference between the GASA, and this additional specialty improved the optimization capability of HGAQSA significantly. Random search in implemented after 10 successive runs of GA and after the each trial the acceptance criteria is hardened. Model parameters of HGAQSA are given in **Table 4-24**.

First Case is resolved for the 1080-Activity projects also. This analysis is repeated three times and HGAQSA obtained global optimum in eight hours in the three trials.

Table 4-24 Model Parameters of the HGAQSA

Genetic Iteration	65
Population Size	150
bitPerActivity	6
Crossover	0,6
Mutation	0,8
Boltzmann Constant	0,7
Random Walk	Project Size$^{3/2}$
Random Walk Period	10

HGAQSA again obtained very good solutions and achieved global optimum at all trials for the 18-Activity, 108-Activity and 360-Activity projects. Analysis duration for the 18-Activity project is 2 seconds, for the 108-Activity duration is 76 seconds and for the 360-Activity project is 24 minutes and 37 seconds.

4.3.4 Analyses by GMASA

Analysis results of GMASA which are given in Table 4-25 are slightly worse than GASA but it can still be considered as satisfactory solution. The model is based on GASA and a memetic search which systematically mutates each activity per run. This property of GMASA increases computational demand but it guarantee visiting every activity.

Model parameters are adjusted in a way to limit the GMASA iterations to a certain level in order to prevent the computational duration increase too fast when the number of activities increases. The model parameters of GMASA is given in Table 4-26.

Table 4-25 Analysis results by GMASA

		Case 1		Case 2	
		Cost	**Duration**	**Cost**	**Duration**
Best Solution	18 Activity	128270	110	127770	126
	108 Activity	772096	660	768168	758
	360 Activity	2596844	2200	2612216	2524
Mean of 10 Analysis	18 Activity	128270	110	127770	126
	108 Activity	773872	660	768549	756
	360 Activity	2601969	2200	2617149	2538

Table 4-26 Model Parameters of the GMASA

GeneticIteration	150
Population	150
bitPerActivity	6
GMASAiteration	5
Boltzmann	0,7
Crossover	0,5
Mutation	0,8
MAperiod	10

4.3.5 Analyses by GASAVNS

GASAVNS gives better results than GMASA but worse than GASA. Analysis results of GASAVNS are shown in Table 4-27. It is seen that the expected improvement could not be obtained by the implemented VNS in this analysis. GASAVNS has high variability in its results especially in the analysis of Case 1 for the 108 Activity projects.

Table 4-27 Analysis results by GASAVNS

		Case 1		Case 2	
		Cost	**Duration**	**Cost**	**Duration**
Best Solution	18 Activity	128270	110	127770	126
	108 Activity	769980	660	766620	756
	360 Activity	2631547	2192	2576662	2524
Mean of 10 Analysis	18 Activity	128270	110	127770	126
	108 Activity	773229	651	766703	756
	360 Activity	2649823	2190	2582936	2528

Model parameters of GASAVNS are given in Table 4-28.

Table 4-28 Model Parameters of the GASAVNS

Iteration	Project Size * 3 + 100
Population Size	Project Size * bitPerActivity
bit/activity	4
SAiteration	$\text{ProjectSize}^{0,4}$
crossover	0,5
mutation	0,8
Maperiod	5
Boltzman	0,5
MaxNeighbor	$\text{ProjectSize}^{0,3}$

4.3.6 Analyses by ACO

Global optimum is obtained by ACO algorithm for the 18-Activity and 108-Activity projects at the analysis of Case 1. In addition to this, global optimum is obtained for the 18-Activity project and near optimum for the 108-Activity project at the analysis of Case 2. Up to this point the analysis results can be considered as optimum, as global optimum is obtained at all of the trials for the 18 and 108-Activity projects of Case 1. However, the analysis results are not satisfactory for the 360-Activity projects which are given in Table 4-29.

Table 4-29 Analysis results by ACO

		Case 1		**Case 2**	
		Cost	**Duration**	**Cost**	**Duration**
Best Solution	18 Activity	128270	110	127770	126
	108 Activity	769620	660	766720	756
	360 Activity	4686194	2312	2556100	2524
Mean of 10 Analysis	18 Activity	128270	110	127870	126,2
	108 Activity	769620	660	768823	757,4
	360 Activity	4902165	2327	2563517	2534

For the Case 1, ACO stuck into local optima and paid liquidated damages. For this reason, results of Case 1 are unacceptable. Results of the 360-Activity project deviate within reasonable limits from global optima. The analysis shows that ACO may get stuck into local optima for the case of large projects with penalty functions. The model parameters of ACO are given in Table 4-30.

Table 4-30 Model Parameters of the ACO

Iteration	200
Population	100
Iteration	5
Max_Pheromone	64
q0	0,2
Alfa	1
Beta	0,001
pheromoneConstant	12
Evaporation	0,7
Best_Evap	0,97

4.3.7 Analyses by PSO

Performance of PSO for the 18-Activity project was very pleasing that it obtained global optima in both Case 1 and Case 2. However, the remaining analysis results are not pleasing which is given in Table 4-31. PSO stuck into local optima for the 108-Activity and 360-Activity projects and pays liquidated damages for the Case 1. If PSO is compared by ACO, it can be said that ACO has better performance since ACO did not pay liquidated damages for the 108-Activity project.

Table 4-31 Analysis results by PSO

		Case 1		**Case 2**	
		Cost	**Duration**	**Cost**	**Duration**
Best Solution	18 Activity	128270	110	127770	126
	108 Activity	2175041	723	768320	761
	360 Activity	2565400	2520	2587670	2537
Mean of 10 Analysis	18 Activity	128270	110	127770	126
	108 Activity	2540167	726,4	768555	762
	360 Activity	10553883	2558	2599934	2543

Analysis results of PSO are not satisfactory for the 360-Activity results. Only in one of the analysis PSO did not paid liquidated damages and paid high amount of liquidated damage in the remaining nine analyses. For the Case 2 PSO give moderate results, only for the 108-Activity project near optimum results are obtained. Model parameters of PSO are given in Table 4-32.

Table 4-32 Model Parameters of the PSO

Iteration	450
Population Size	500
Random Iteration	9
Resolution	64
c1	2
c2	2
Vmax	16

4.3.8 Analyses by EMS

Results of EMS are only better than GA and worse than the remaining meta-heuristic algorithms. EMS obtains global optimum in none of the analyses. Only in 18-Activity projects near optimum results are obtained. Schedules with liquidated damages are obtained for the 108 and 360-Activity projects. Model parameter of EMS is only the population size and random iteration. Population size is taken as twice of the project size and random iteration is taken as five.

Table 4-33 Analysis results by EMS

		Case 1		**Case 2**	
		Cost	**Duration**	**Cost**	**Duration**
Best Solution	18 Activity	129670	110	128650	128
	108 Activity	996368	676	841255	812
	360 Activity	5570200	2349	2994425	2987
Mean of 10 Analysis	18 Activity	139704	109,5	130068	131
	108 Activity	1141976	684	867222	821
	360 Activity	5586985	2250,7	3045552	3002

4.3.9 Conclusion

Characteristics of 18-Activity project significantly differ from the 7-activity project. First of all, trial and error solution of 18-activity project is almost impossible even by a high speed computer. A small increase in the project size ends up with a high amount of increase in the number of possible project combinations. This is a sound example for the illustration of the combinatorial expansion of TCT type problems.

Abrupt changes are observed in the performances of the meta-heuristic algorithms. Optimum solution of 7-activity project is obtained by all of the algorithms. However, GA and ESS could not obtain global optima in any of the trials in 18-activity project. HGAQSA performed the best performance and had obtained global optimum at all of the trials within reasonable computation duration. Hybrid local and global search capability of HGAQSA is the main advantage of the meta-heuristic algorithm for the convergence of global optima.

GASA has the second rank for the obtaining of global optima. The main disadvantage of GASA when compared with HGAQSA is its slow convergence. However, GASA also obtains global optimum even for the large projects within reasonable computation time.

After the GASA, ACO can be considered as the third successful meta-heuristic algorithm. Although ACO does not represent satisfactory results, it gives pleasing results for the small and medium-sized projects.

GASAVNS, GMASA and PSO give successful results for the small-size project but did not give satisfactory results for the medium and large-projects. Finally, GA and EMS do not give pleasing results in none of the analyses.

4.4 TCT Analysis with 29 Activity Project

Test problem which is previously analyzed by Chassiakos and Sakellaropoulos (2005) is analyzed. The time cost trade-off optimization problem consists of a precedence diagram with 29 activities and various crashing alternatives for a highway upgrading project. Project's logical restrictions are shown in Table 4-34 and crashing alternatives are illustrated in Table 4-35. Indirect project cost is determined as €1200/day. In the original project, there were additional restrictions on some of the start and finish times of the activities. The restrictions are omitted in order to make the duplication of the project possible.

In the study of Chassiakos and Sakellaropoulos (2005), project is analyzed with an additional case besides the constant overhead cost. A penalty of €1500/day is paid if the project is completed later or a bonus of €500/day is gained if the project is completed earlier than 240 days. Two contract cases are analyzed and the results are compared with the results published by Chassiakos and Sakellaropoulos (2005).

29-activity project consists of 6 single mode, 6 two-mode and 17 three-mode activities. Number of combinations is computed as $2^6 * 3^{17}$ which is 8.264.970.432. This number is almost twice of the combination of the 19-activity project. The project is analyzed by GA, GASA, HGASA, GMASA, GASAVNS, ACO, PSO and ESS meta-heuristic algorithms.

Table 4-34 Logical restrictions of the 29-Activity project

Activity ID	**Activity Description**	**Precedence relations**
	Service road A	
1	Road excavation	NULL
2	Embankment construction	1SS+5, 1FF
3	Subbase and base layers	1SS+10, 1FF+3, 2SS + 10, 2FF+3
4	Asphalt layer	3FS
5	Temporary marking and signing	4FS
	Service road B	
6	Earth and semirock excavation	1FS
7	Embankment construction	6SS, 6FF
8	Subbase and base layers	7FS
9	Asphalt layer	4FS, 8FS
10	Temporary marking and signing	5FS, 9FS
	Main Road	
11	Traffic diversion	5FS, 10FS
12	Rock excavation	11FS
13	Earth and semirock excavation - existing pavement removal	12SS+6
14	Subgrade stabilization, retaining wall/culvert construction	12FF+8, 13FF+8
15	Embankment construction	12SS+6, 12FF, 14FS-6, 14FF+10
16	Drainage layer	15SS+10, 15FF
17	Drainage pipe construction	15FS-10, 15FF
18	Electrical ins. at roadway verges	15FS
19	Planting at roadway verges	15FS+12
20	Ditches	17SS+10, 17FF+5
21	Subase layer	16FS-10, 16FF, 20SS+12, 20FF+2
22	Base layer	21SS+15, 21FF+2
23	Median island (New Jersey)	22FS-15, 22FF
24	Electrical ins. at median island	23SS+15, 23FF+5
25	Asphalt layer No. 1	23FS-10, 23FF
26	Asphalt layer No. 2	25FS-10, 25FF
27	Friction course overlay	26FS-10, 26FF
28	Final marking and signing	27FS-10, 27FF
29	Traffic restoration	18FS, 19FS, 24FS, 28FS

Crashing options of the activities are shown in Table 4-35. Cost of the activities is given in €1000.

Table 4-35 Crashing alternatives of the 29-Activity project

Activity ID	**Normal**		**Alternative 1**		**Alternative 2**	
	Time	**Cost**	**Time**	**Cost**	**Time**	**Cost**
1	15	60	12	68	-	-
2	25	30	20	38	15	44
3	25	50	20	54	15	60
4	12	17	9	21	-	-
5	6	3	-	-	-	-
6	12	27	9	32	-	-
7	6	8	-	-	-	-
8	20	44	15	48	12	54
9	12	15	9	22	-	-
10	6	3	-	-	-	-
11	1	0,5	-	-	-	-
12	25	95	20	105	15	109
13	15	34	12	41	9	51
14	12	9	9	13	-	-
15	25	30	15	38	12	42
16	40	78	35	85	30	90
17	25	23	20	26	12	35
18	20	14	15	18	12	24
19	25	14	20	19	15	24
20	20	38	15	42	-	-
21	40	42	35	50	30	58
22	40	36	30	48	25	56
23	40	65	35	74	25	79
24	9	7	-	-	-	-
25	25	45	20	51	15	59
26	25	50	20	58	15	64
27	30	60	25	72	20	78
28	12	9	9	13	7	18
29	1	0,5	-	-	-	-

In the first case there are four different optimum solutions that give the same total project cost. Optimum solution of 241 days of project duration is given in Table 4-36.

Table 4-36 Schedule of 241 days project duration

Activity ID	ES	EF	Cost	Duration
1	0	15	60000	15
2	5	30	30000	25
3	15	35	54000	20
4	35	47	17000	12
5	47	53	3000	6
6	15	27	27000	12
7	21	27	8000	6
8	27	47	44000	20
9	47	59	15000	12
10	59	65	3000	6
11	65	66	500	1
12	66	91	95000	25
13	72	87	34000	15
14	87	99	9000	12
15	93	118	30000	25
16	103	143	78000	40
17	108	133	23000	25
18	118	138	14000	20
19	130	155	14000	25
20	118	138	38000	20
21	133	173	42000	40
22	148	178	48000	30
23	163	188	79000	25
24	184	193	7000	9
25	178	203	45000	25
26	193	218	50000	25
27	208	238	60000	30
28	228	240	9000	12
29	240	241	500	1

Direct cost of the project is €937000 and the indirect cost is €289200 which makes the total project cost €1226200. Optimum solution of 246 days of project duration is given in Table 4-37.

Table 4-37 Schedule of 246 days project duration

Activity ID	ES	EF	Cost	Duration
1	0	15	60000	15
2	5	30	30000	25
3	15	35	54000	20
4	35	47	17000	12
5	47	53	3000	6
6	15	27	27000	12
7	21	27	8000	6
8	27	47	44000	20
9	47	59	15000	12
10	59	65	3000	6
11	65	66	500	1
12	66	91	95000	25
13	72	87	34000	15
14	87	99	9000	12
15	93	118	30000	25
16	103	143	78000	40
17	108	133	23000	25
18	118	138	14000	20
19	130	155	14000	25
20	118	138	38000	20
21	133	173	42000	40
22	148	188	36000	40
23	173	198	79000	25
24	194	203	7000	9
25	188	208	51000	20
26	198	223	50000	25
27	213	243	60000	30
28	233	245	9000	12
29	245	246	500	1

First case of the 29-Activity project has four different solutions which have the same total project cost. In Table 4-38 analysis results of meta-heuristic algorithms are given. The results include the 10 trials per each meta-heuristic algorithm. By examining the results it can be concluded that HGAQSA is the best meta-heuristic algorithm because HGAQSA converged to global optima at all of the trials in shortest analyses time. Besides HGAQSA, GASA and GMASA converged to global optima at all of the trials with slightly longer analyses duration. The three algorithms randomly converged into one of the four global optima.

Table 4-38 Analysis results for only overhead cost

	GA		ACO		PSO		GASAVNS		EMS		HGAQSA		GMASA		GASA	
	Cost	**Dur**	**Cost**	**Dur**	**Cost**	**Dur**	**Cost**	**Dur**	**Cost**	**Dur**	**Cost**	**Dur**	**Cost**	**Dur**	**Cost**	**Dur**
1	1230200	266	1228200	246	1226200	251	1226200	251	1244200	221	1226200	251	1226200	246	1226200	241
2	1228200	241	1228200	251	1228600	253	1226200	251	1245600	233	1226200	241	1226200	246	1226200	246
3	1230200	266	1226200	251	1226200	241	1226200	246	1237000	255	1226200	236	1226200	246	1226200	241
4	1228200	246	1228200	256	1226200	241	1226200	251	1242800	254	1226200	251	1226200	241	1226200	246
5	1230200	266	1228200	256	1226200	246	1228200	236	1248600	228	1226200	241	1226200	241	1226200	246
6	1228200	256	1230200	261	1226200	236	1226200	241	1241600	238	1226200	241	1226200	236	1226200	241
7	1228200	256	1228200	256	1226200	251	1230200	266	1247200	251	1226200	236	1226200	241	1226200	236
8	1228200	256	1228200	256	1230600	243	1226200	236	1238600	238	1226200	251	1226200	241	1226200	241
9	1228200	251	1228200	246	1226200	246	1228200	246	1245600	238	1226200	236	1226200	236	1226200	246
10	1226200	251	1228200	246	1226200	241	1229800	224	1239800	264	1226200	251	1226200	246	1226200	236

Table 4-39 Analysis results with overhead, penalty and bonus

	GA		ACO		PSO		GASAVNS		EMS		HGAQSA		GMASA		GASA	
	Cost	**Dur**	**Cost**	**Dur**	**Cost**	**Dur**	**Cost**	**Dur**	**Cost**	**Dur**	**Total Cost**	**Dur**	**Total Cost**	**Dur**	**Total Cost**	**Dur**
1	1257700	221	1247400	232	1220700	221	1220700	221	1240600	218	1220700	221	1220700	221	1220700	221
2	1226700	211	1236100	233	1220700	221	1220700	221	1237900	207	1220700	221	1220700	221	1220700	221
3	1246000	200	1238100	243	1222600	218	1220700	221	1239700	231	1220700	221	1220700	221	1220700	221
4	1252100	213	1230200	236	1221700	211	1223200	216	1238400	222	1220700	221	1220700	221	1220700	221
5	1244000	220	1234700	221	1223600	218	1220700	221	1237100	213	1220700	221	1220700	221	1220700	221
6	1233900	217	1224100	223	1223600	218	1220700	221	1232400	212	1220700	221	1220700	221	1220700	221
7	1245800	224	1235800	224	1220700	221	1220700	221	1235000	210	1220700	221	1220700	221	1220700	221
8	1237200	226	1230000	230	1225400	212	1221200	226	1241800	204	1220700	221	1220700	221	1220700	221
9	1242000	210	1231200	226	1221700	211	1220700	221	1233600	228	1220700	221	1220700	221	1220700	221
10	1237800	214	1230000	230	1224600	208	1222300	209	1238200	236	1220700	221	1220700	221	1220700	221

GASAVNS, PSO and ACO converged into global optima at some of the analysis. The algorithms had converged to near optimum solutions at the remaining analysis within reasonable computation duration. GA and EMS could not converge into global optima in none of the analysis.

Direct cost of the project is €931000 and the indirect cost is €295200 which makes the total cost €1226200. Optimum schedule of project duration of 236 days is given in Table 4-40.

Table 4-40 Schedule of 236 days project duration

Activity ID	ES	EF	Cost	Duration
1	0	15	60000	15
2	5	30	30000	25
3	15	35	54000	20
4	35	47	17000	12
5	47	53	3000	6
6	15	27	27000	12
7	21	27	8000	6
8	27	47	44000	20
9	47	59	15000	12
10	59	65	3000	6
11	65	66	500	1
12	66	91	95000	25
13	72	87	34000	15
14	87	99	9000	12
15	93	118	30000	25
16	103	143	78000	40
17	108	133	23000	25
18	118	138	14000	20
19	130	155	14000	25
20	118	138	38000	20
21	133	173	42000	40
22	148	178	48000	30
23	163	188	79000	25
24	184	193	7000	9
25	178	198	51000	20
26	188	213	50000	25
27	203	233	60000	30
28	223	235	9000	12
29	235	236	500	1

Direct cost of the third optimum solution is €943000 and the indirect cost is €283200 which makes the total cost €1226200. Optimum schedule of project duration of 251 days is given in Table 4-41.

Direct cost of the fourth optimum solution is €925000 and the indirect cost is €301200 which makes the total cost €1226200.

Table 4-41 Schedule of 251 days project duration

Activity ID	**ES**	**EF**	**Cost**	**Duration**
1	0	15	60000	15
2	5	30	30000	25
3	15	35	54000	20
4	35	47	17000	12
5	47	53	3000	6
6	15	27	27000	12
7	21	27	8000	6
8	27	47	44000	20
9	47	59	15000	12
10	59	65	3000	6
11	65	66	500	1
12	66	91	95000	25
13	72	87	34000	15
14	87	99	9000	12
15	93	118	30000	25
16	103	143	78000	40
17	108	133	23000	25
18	118	138	14000	20
19	130	155	14000	25
20	118	138	38000	20
21	133	173	42000	40
22	148	188	36000	40
23	173	198	79000	25
24	194	203	7000	9
25	188	213	45000	25
26	203	228	50000	25
27	218	248	60000	30
28	238	250	9000	12
29	250	251	500	1

Table 4-42 Optimum schedule of the second case

Activity ID	ES	EF	Cost	Duration
1	0	15	60000	15
2	5	30	30000	25
3	15	35	54000	20
4	35	47	17000	12
5	47	53	3000	6
6	15	27	27000	12
7	21	27	8000	6
8	27	47	44000	20
9	47	59	15000	12
10	59	65	3000	6
11	65	66	500	1
12	66	91	95000	25
13	72	87	34000	15
14	87	99	9000	12
15	93	118	30000	25
16	103	143	78000	40
17	108	133	23000	25
18	118	138	14000	20
19	130	155	14000	25
20	118	138	38000	20
21	133	173	42000	40
22	148	178	48000	30
23	163	188	79000	25
24	184	193	7000	9
25	178	193	59000	15
26	183	198	64000	15
27	188	218	60000	30
28	208	220	9000	12
29	220	221	500	1

Second case which has liquidated damage for late project completion and bonus for early completion is analyzed in the same manner and the analyses results are given in Table 4-39. The results of the second case are very similar to the first case. HGAQSA, GMASA and GASA converge into global optimum at all trials. GASAVNS and PSO converge into global optima in some of the trials but obtained near optimum results. GA, ACO and EMS could not obtain optimum solution in none of the trials. Apart from ACO, all meta-heuristic algorithms end up with the more or less same quality results. The reason of failure to obtain global optima in Case 2 can be the contract condition with liquidated damages. Optimum schedule of project duration for the second case is given in Table 4-42.

Direct cost of the optimum solution is €965000 and the indirect cost is €265200. However, since the project is completed 19 days before the contract requirement €9500 bonus is gained which makes the total cost €1220700.

When the analysis results are compared with the solutions obtained by Chassiakos and Sakellaropoulos (2005), it is seen that the solution obtained by meta-heuristic algorithms give better results by €5200 for the first case and by €3200 for the second case. This is because of the removed restrictions on some of the start and finish times of some of the activities. It is not possible to compare the schedule since the authors did not give the schedule in tabular form.

The 29-Activity problem is different than the previous two projects in which it has complicated CPM logical restrictions where activities have more than one restriction with positive or negative lags. The problem is previously solved by Chassiakos and Sakellaropoulos (2005) by LP/IP. The authors had 265 constraints and 129 variables to solve. The genetic representation of this problem has only 23 parameters to solve which took only 2 seconds to solve by HGAQSA.

The analyses of 29-Actitivity project show that apart from simple FS relationships the meta-heuristic algorithms have the capability of solving TCT problems with complicated logical relationships.

Analysis duration is given in Table 4-43. Accept for GA and EMS, the meta-heuristic algorithms have reasonable computation duration. Although EMS and GA have the longest computation duration, they could not obtain global optimum.

Table 4-43 Analysis duration of one trial of meta-heuristic algorithms.

GA	24 Sec
ACO	8 Sec
PSO	8 Sec
GMASA	8 Sec
HGAQSA	2 Sec
GASAVNS	8 Sec
EMS	67 Sec
GASA	5 Sec

In order to test the algorithms on a more difficult problem, the 29-activity project is duplicated nine times and 290-activity project is obtained. Optimum solutions of first case is €12.262.000 and second case is €12.207.000. Algorithms are stopped at the end of the one millionth evaluation. The analyses are repeated ten times and mean of the ten trials are shown in Table 4-44 and Table 4-45.

Table 4-44 10X 29-activity project analysis results of first case

Algorithm	**1000**	**2500**	**5000**	**10000**	**50000**	**250000**	**1000000**
GA	12900160	12900160	12900160	12897080	12894860	12892060	12879680
GASA	12912920	12912620	12911120	12908520	12893600	12781680	12475420
HGAQSA	12363440	12363440	12361700	12359260	12353300	12318200	12299220
GMASA	12901640	12901640	12900300	12892560	12872460	12850720	12837080
GASAVNS	12934640	12926380	12920900	12918720	12865180	12685980	12405920
PSO	12503300	12486020	12478740	12468940	12467820	12467680	12467680
ACO	12687540	12661660	12652400	12636060	12596400	12566000	12492000
ESS	12965020	12965020	12965020	12965020	12963620	12933660	12912340

Table 4-45 10X 29-activity project analysis results of second case

Algorithm	**1000**	**2500**	**5000**	**10000**	**50000**	**250000**	**1000000**
GA	12825700	12825700	12825700	12824670	12820490	12813600	12802140
GASA	12843200	12842040	12841530	12836460	12816390	12692420	12379630
HGAQSA	12269800	12269800	12269800	12269800	12259370	12225530	12208820
GMASA	12821760	12821110	12820110	12806310	12787500	12783030	12753800
GASAVNS	12853260	12851820	12841860	12833590	12778530	12591450	12304340
PSO	12442700	12422250	12411390	12408130	12407890	12407880	12407880
ACO	12893920	12857710	12846110	12728060	12627460	12525070	12367390
ESS	12883730	12883150	12883150	12882600	12879000	12856690	12836520

29-activity project is an important example in which it includes SS and FF type logical relationships and contains lags in some of these relationships. This example project is a good candidate to show the capability of the scheduling program. In addition to this, 29-activity project contains activities with only single activity execution mode. As this is the case, number of parameters to solve is not equal to the number of activities of the project. 29-activity project has 23 parameters to be determined. These are the two important characteristics of the project.

HGAQSA, GMASA and GASA obtained global optimum in all of the trials, both for the first and the second cases. HGAQSA is the most successful algorithm in this example since it had obtained the global optimum in the shortest time. GASA and GMASA are also successful algorithms. However, in the previous analysis it is seen that their convergence are relatively slow. In this analysis, two algorithms obtained global optima. From this result, it can be inferred that the number of local minima is relatively low as the slow converging algorithm GMASA obtained global optima in all of the trials.

GASAVNS and PSO present similar results that the two algorithms obtained global optima in some of the analysis and converged to near-optimum solutions in the rest of the analysis. GASAVNS is known to be an optimum converging algorithm. Similar to GMASA, converging speed of GASAVNS is slow and 150.000 evaluations are not enough for GASAVNS to converge into global optimum. On the other hand, PSO is known to be a fast near-optimum converging algorithm and its convergence rate decreases at the further iterations. PSO showed similar trend in this analysis.

ACO does not present satisfactory results in the analysis of 29-activity project. The algorithm obtained global optimum only once and the analysis results are not close to the global optimum especially for the second case.

Although analysis duration of GA and ESS are the longest, worst results are obtained by these two algorithms. Two algorithms are failed to present proper results.

The 29-activity project is duplicated nine times and 290-activity project is obtained. The project is again analyzed for the two contract conditions with stopping criteria of at most 1 million project evaluations. Highly challenging test problem is obtained with this method. Since the solution of 29-activity project is known, optimum solution of 290-activity project is also known. The aim of this analysis is both to measure the convergence ability and convergence rate of the meta-heuristic algorithms.

At the end of the 1 millionth evaluation, no meta-heuristic algorithm were able to obtain global optimum. HGAQSA present best performance and obtain the most successful solution. GASA, PSO, GASAVNS and ACO followed HGAQSA. However, there is significant difference between the results of HGAQSA and the four meta-heuristic algorithms. The slow convergence of GMASA is apparently seen in this analysis, where the algorithm is far from obtaining a near-optimum solution. Finally, GA and ESS could not obtain successful results.

4.5 TCT Analysis with 63 Activity Project

Previous projects analyzed were small sized projects with at most 8.3 Billion schedule alternatives. The difficulties of analyses are not as same as with the real cases. 18-Activity and 29-activity projects are duplicated and medium and large-sized projects are obtained; however the duplicated projects do not have complicated logical relationships as same-sized project should have. As a result the analysis can not be considered as the real solution of large-sized projects.

In order to measure the performance of the meta-heuristic algorithms on the medium sized projects, a 63-Activity hypothetic project which is given in Table 4-46 is generated.

The project has only Finish-to-Start logical relationships with no positive or negative lag. Crashing alternatives are given in Cost & Duration columns. Minimum crashing alternative that an activity has is three and maximum crashing alternative number that an activity has is five.

63-activity project consists of 2 3-crashing option activity, 15 4-crashing option activity and 46 5-crashing option activity. As a result, total number of possible schedule combinations of the 63-activity project is computed as $3^2 * 4^{15} * 5^{46} = 1{,}373*10^{42}$.

Table 4-46 63-Activity Hypothetical project

Activity ID	Pred	Method 1		Method 2		Method 3		Method 4		Method 5	
		Cost	Dur	Cost	Dur	Cost	Dur	Cost	Dur	Cost	Dur
1	NULL	3750	14	4250	12	5400	10	6250	9	-	-
2	NULL	11250	21	14800	18	16200	17	19650	15	-	-
3	NULL	22450	24	24900	22	27950	19	31650	17	-	-
4	NULL	17800	19	19400	17	21600	15	-	-	-	-
5	NULL	31180	28	34200	26	38250	23	41400	21	-	-
6	1	54260	44	58450	42	63225	38	68150	35	-	-
7	1	47600	39	50750	36	54800	33	59750	30	-	-
8	2	62140	52	69700	47	72600	44	81750	39	-	-
9	3	72750	63	79450	59	86250	55	91500	51	99500	49
10	4	66500	57	70250	53	75800	50	80750	46	86450	41
11	5	83100	63	89450	59	97800	55	104250	50	112400	45
12	6	75500	68	82000	62	87500	58	91800	53	96550	49
13	7	34250	40	38500	37	43950	33	48750	31	-	-
14	8	52750	33	58450	30	63400	27	66250	25	-	-
15	9	38140	47	41500	40	47650	35	54100	32	-	-
16	9, 10	94600	75	101250	70	112750	66	124500	61	132850	57
17	10	78450	60	84500	55	91250	49	94640	47	-	-
18	10, 11	127150	81	143250	73	154600	66	161900	61	-	-
19	11	82500	36	94800	34	101700	30	-	-	-	-
20	12	48350	41	53250	37	59450	34	66800	32	-	-
21	13	85250	64	92600	60	99800	57	107500	53	113750	49
22	14	74250	58	79100	53	86700	50	91500	47	97400	42
23	15	66450	43	69800	41	75800	37	81400	33	88450	30
24	16	72500	66	78500	62	83700	58	89350	53	96400	49
25	17	66650	54	70100	50	74800	47	79500	43	86800	40
26	18	93500	84	102500	79	111250	73	119750	68	128500	62
27	20	78500	67	86450	60	89100	57	91500	56	94750	53
28	21	85000	66	89750	63	92500	60	96800	58	100500	54
29	22	92700	76	98500	71	104600	67	109900	64	115600	60
30	23	27500	34	29800	32	31750	29	33800	27	36200	26
31	19, 25	145000	96	154800	89	168650	83	179500	77	189100	72
32	26	43150	43	48300	40	51450	37	54600	35	61450	33
33	26	61250	52	64350	49	68750	44	74500	41	79500	38
34	28, 30	89250	74	93800	71	99750	66	105100	62	114250	57
35	24,27,29	183000	138	201500	126	238000	115	283750	103	297500	98
36	24	47500	54	50750	49	56800	42	62750	38	68250	33
37	31	22500	34	24100	32	26750	29	29800	27	31600	24
38	32	61250	51	65800	47	71250	44	76500	41	80400	38
39	33	81150	67	87600	61	92100	57	97450	52	102800	49
40	34	45250	41	48400	39	51200	36	54700	33	58200	31
41	35	17500	37	21200	31	26850	27	32300	23	-	-
42	36	36400	44	39750	41	42800	38	48300	32	50250	30
43	36	66800	75	71200	69	76400	63	81300	59	86200	54
44	37	102750	82	109500	76	127000	70	136800	66	146000	63

Table 4-46 Continued

45	39	84750	59	91400	55	101300	51	126500	47	142750	43
46	39	94250	66	99500	63	108250	59	118500	55	136000	50
47	40	73500	54	78500	51	83600	47	88700	44	93400	41
48	42	36750	41	39800	39	43800	37	48500	34	53950	31
49	38,41,44	267500	173	289700	159	312000	147	352500	138	397750	121
50	45	47800	101	61300	74	76800	63	91500	49	-	-
51	46	84600	83	93650	77	98500	72	104600	65	113200	61
52	47	23150	31	27600	28	29800	26	32750	24	35200	21
53	43, 48	31500	39	34250	36	37800	33	41250	29	44600	26
54	49	16500	23	17800	22	19750	21	21200	20	24300	18
55	52, 53	23400	29	25250	27	26900	26	29400	24	32500	22
56	50, 53	41250	38	44650	35	47800	33	51400	31	55450	29
57	51, 54	37800	41	41250	38	45600	35	49750	32	53400	30
58	52	12500	24	13600	22	15250	20	16800	18	19450	16
59	55	34600	27	37500	24	41250	22	46750	19	50750	17
60	56	28500	31	30500	29	33250	27	38000	25	43800	21
61	56, 57	22500	29	24750	27	27250	25	29800	22	33500	20
62	60	38750	25	41200	23	44750	21	49800	19	51100	17
63	61	9500	27	9700	26	10100	25	10800	24	12700	22

Number of combinations is significantly large that it makes totally impossible to perform exhaustive enumeration even with parallel computing. As this is the case, importance of convergence capability of optimum search algorithm becomes significant.

63-Activity project is analyzed by the 8 meta-heuristic algorithms and the results shown in Table 4-47 and Table 4-48 are obtained. The 63-Activity project is analyzed for two different conditions. First case is $2300 constant daily overhead cost and the second case is $3500 daily overhead cost. The optimum solution of this problem is $5421120 and $6176170 for the first and second cases respectively.

Optimum solution of the 63-activity project is obtained by commercial optimization software AIMMS. AIMMS is linear, integer programming optimization software. The computational duration of 63-activity project was only 0.05 seconds which is excessively shorter duration when compared with the analysis duration of meta-heuristic algorithms.

The analysis results are slightly different from the previous analyses that HGAQSA could not give the best results. GMASA and GASAVNS obtain the global optimum in all of the trials in the first case. HGAQSA obtains global optimum in 7 trials out of 10 and the remaining three trials are close to global optimum. Finally GASA can be considered as a successful meta-heuristic algorithm although it could not obtain global optimum in any of the trials. However, the analysis results are close to global optimum and deviation of the results is not high. PSO and ACO also give satisfactory results but they are not as good as the previous methods results. In addition to this deviation of ACO is slightly high. Finally EMS and GA are the worst meta-heuristic methods of the case 1.

Case 2 is slightly more difficult problem than Case 1 which is easy to induct if the analyses results of both cases are examined. Only GMASA could obtain global optimum in all trials. GASAVNS obtained global optimum in 5 trials out of 10. HGAQSA could not obtain global optima in any of the trials, however near optimum solutions are obtained in all trials. Stable results are also obtained by GASA but in Case 2 the results are a little further from the global optima but within acceptable region. Similarly PSO and ACO give satisfactory results but they are not as successful as GASA. Finally, GA and ESS failed to give proper schedule for the TCT problem of 63-Activity project.

Both Case 1 and Case 2 are repeated by duplicating the 63-Activity project 9 times. By duplicating the 63-Activity project, 630-Activity project is obtained and the project is analyzed 10 times with the conditions of case 1 and case 2. The mean of the 10 analysis of 630-activity project is given in Table 4-49 and Table 4-50.

Global optimum of the case 1 is \$54211200 and case 2 is \$61761700. Meta-heuristic algorithms could not obtain global optima in any of the trials. The reason of this can be explained by the difficulty of the problem and the early stopping condition.

HGAQSA, PSO and GASAVNS successfully converged into near optimum solutions. GMASA is known to be slow converging algorithm and it is expected that in further iterations the algorithm would obtain near optimum solutions. On the other hand, GASA could not perform well when compared with its previous results. ESS and GA are again the worst meta-heuristic algorithms among the others.

Especially PSO has very high rate of convergence to near optimum solutions. However, the algorithm gets stuck into local optima and could not improve its early solution considerably. However, HGAQSA could not initially converge near optimum solutions as fast as PSO, but HGAQSA obtains better solutions at later iterations.

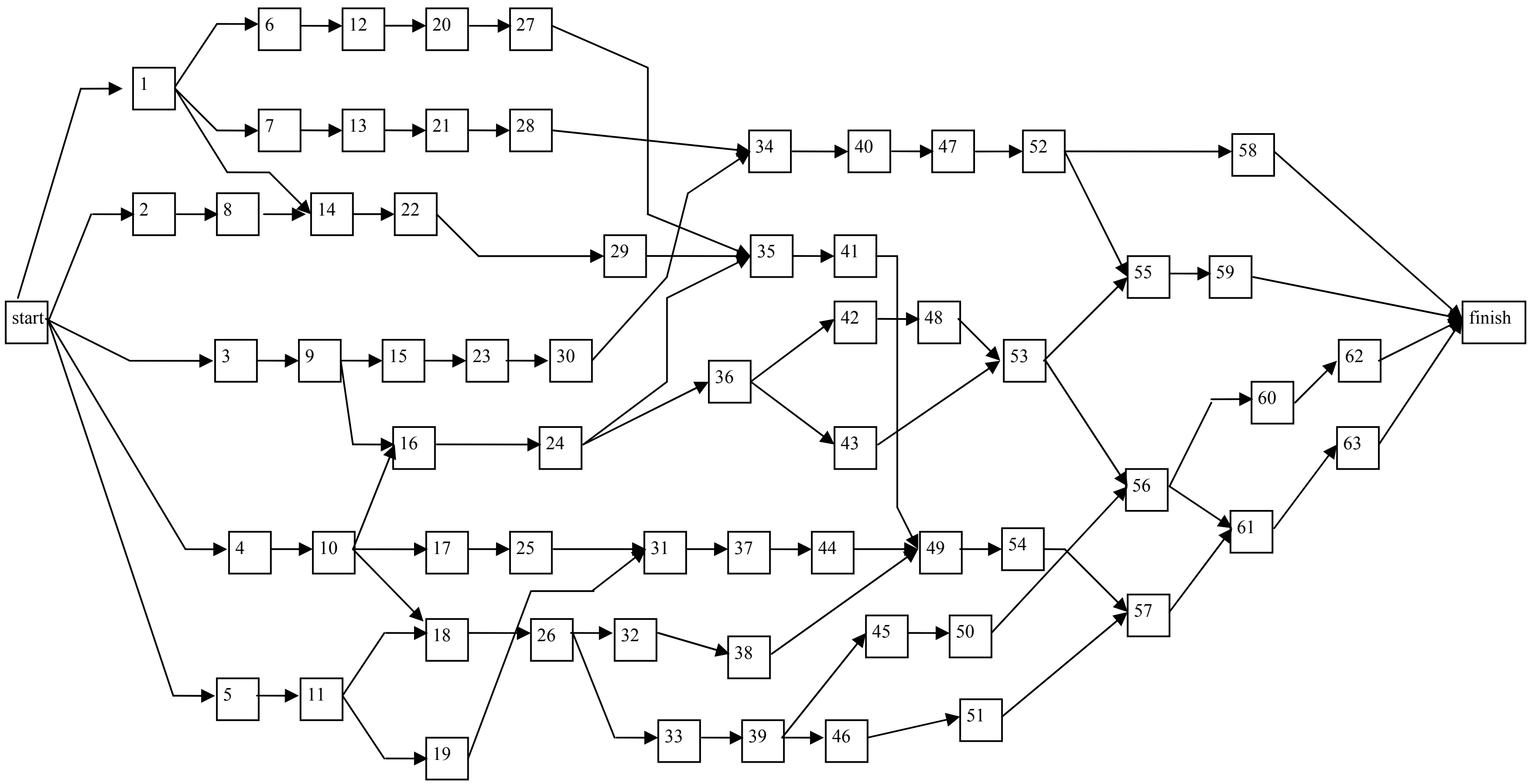

Figure 4.4 AoN diagram of 63 activity project

Table 4-47 Analysis results of 63-Activity project for the Case 1

Analysis No	GMASA	Dur	HGAQSA	Dur	GASAVNS	Dur	ACO	Dur	EMS	Dur	PSO	Dur	GASA	Dur	GA	Dur
1	5421120	630	5421120	630	5421120	630	5490120	635	5537190	617	5421620	637	5421320	633	5704200	641
2	5421120	630	5421120	630	5421120	630	5494410	653	5548930	631	5428920	644	5421320	633	5712485	661
3	5421120	630	5421320	633	5421120	630	5491180	638	5532710	628	5439620	651	5421620	633	5722260	650
4	5421120	630	5421120	630	5421120	630	5491620	657	5530750	647	5422920	634	5421320	633	5713450	653
5	5421120	630	5421320	633	5421120	630	5494920	644	5527270	621	5440570	651	5421620	633	5699650	645
6	5421120	630	5421320	633	5421120	630	5486630	626	5526450	633	5421320	633	5421620	633	5684295	639
7	5421120	630	5421120	630	5421120	630	5495080	664	5538750	620	5421320	633	5421620	633	5695655	640
8	5421120	630	5421320	633	5421120	630	5490350	661	5551255	636	5421620	633	5421620	633	5707600	621
9	5421120	630	5421120	630	5421120	630	5490680	643	5538340	624	5421320	633	5421620	633	5693015	641
10	5421120	630	5421120	630	5421120	630	5492210	635	5532920	622	5421320	633	5421620	633	5690790	623

Table 4-48 Analysis results of 63-Activity project for the Case 2

Analysis No	GMASA	Dur	HGAQSA	Dur	GASAVNS	Dur	ACO	Dur	EMS	Dur	PSO	Dur	GASA	Dur	GA	Dur
1	6176170	621	6178670	629	6176170	621	6219220	631	6286080	612	6201720	644	6181270	629	6462580	617
2	6176170	621	6178570	629	6176170	621	6205850	632	6329050	601	6217470	629	6177570	630	6411540	651
3	6176170	621	6179470	621	6176220	621	6234520	626	6316480	616	6210170	644	6184670	633	6442440	647
4	6176170	621	6178570	630	6176170	621	6223830	640	6340190	607	6218170	648	6183320	631	6420500	639
5	6176170	621	6177820	630	6176170	621	6231440	617	6333790	593	6216020	649	6180420	618	6447900	648
6	6176170	621	6178270	630	6176520	621	6197070	627	6339620	623	6207870	647	6180520	629	6433810	627
7	6176170	621	6179320	621	6176520	621	6247850	604	6336680	600	6216220	651	6179870	629	6439240	618
8	6176170	621	6178270	630	6176170	621	6231860	635	6327695	600	6215420	649	6180620	621	6449790	623
9	6176170	621	6178570	630	6176520	621	6198650	623	6336290	594	6208920	645	6177270	629	6443805	630
10	6176170	621	6178070	618	6176670	621	6262830	651	6297120	620	6198520	642	6182020	630	6450065	629

The 630-activity project was the most difficult project and as a result the best solution deviated more than 2% from the global optimum. One analysis took approximately 73 minutes which can be elongated if more precise solutions are needed.

In the analysis the project is scheduled 1 million times in order to limit the computation duration within reasonable limits and to measure the convergence capability of the algorithms to the near optimum solution. Schedule counter is increased by one after evaluation of a schedule. Therefore, for the GA based meta-heuristic algorithms although the mutation is rejected, the rejected mutation is also counted.

Table 4-49 Mean values of 10 run of 630-Activity project of Case 1

Algorithm	1000	2500	5000	10000	50000	250000	1000000
GA	59006710	59005575	59005575	59005575	58996707	58983147	58973912
GASA	59036652	59032161	59029116	59021906	59004968	58752744	57344739
HGAQSA	55519922	55519922	55519922	55519922	55515989	55450532	55257834
GMASA	58610138	58609988	58609738	58604393	58560383	58467643	58238341
GASAVNS	58632095	58627051	58621995	58608931	58531434	58050451	57209157
PSO	56605465	54821050	54816150	54815880	54815790	54815790	54815125
ACO	57137808	57074624	57024915	57001015	56896886	56703583	56506595
ESS	59018710	59018698	59017710	59017544	59015310	59014429	59007254

Table 4-50 Mean values of 10 run of 630-Activity project of Case 2

Algorithm	1000	2500	5000	10000	50000	250000	1000000
GA	66404772	66404633	66404630	66402790	66395840	66364424	66350018
GASA	66516795	66509778	66505487	66497151	66451864	66283385	65194541
HGAQSA	63293505	63293505	63293505	63293505	63288277	63236344	63046332
GMASA	66471550	66471550	66471550	66471550	66433571	66343911	66216394
GASAVNS	66470339	66439558	66424236	66409958	66316913	65892994	65019518
PSO	64701615	63170076	63144015	63129480	63121500	63121500	63121500
ACO	64780611	64718752	64682565	64672048	64574989	64390384	64219993
ESS	66519873	66518756	66518356	66517218	66517218	66514726	66509967

HGAQSA and PSO can be suggested as good candidates for the solution of TCT problems. If there would be a choice to be made between the two, HGAQSA can be chosen. However, if the project size is large and near-optimum solutions are considered as satisfactory, than PSO can be a good candidate.

Optimum solution of TCT type problems are guaranteed by linear integer programming algorithms. Any kind of simple TCT type problem can easily be solved by commercial linear programming based optimization software. As this is the case, the endeavor of solving simple TCT problems by meta-heuristic algorithm can be seen meaningless. However, TCT problems are good candidates to measure the relative performances of the meta-heuristic algorithms. The performances of the algorithms are compared and significant improvements are obtained by adaptations on the algorithm.

Although the memory capacity of the computers is high and still increasing, there are still some limitations for the solution of large project sizes. The 63-activity project required more than 64 MB of memory for the optimum solution. It is known that, memory requirement of the integer-linear programming increases proportional with the square of the project size. As a result of this, for the solution of large projects, there might be limitations due to the memory limitations of the computers.

In addition to this, ESS and ACO meta-heuristic algorithms could not adopted for the analyses of resource leveling problem. If the algorithms are not tested on TCT type problems, than the comparison of GA based methods with ACO and ESS could not be done.

By using meta-heuristic algorithms, the largest project analyzed was only 18-activity project. In this thesis, TCT problem is analyzed by 63-activity project and its global optimum is found by meta-heuristic algorithms. This is a notable improvement on the meta-heuristic algorithms.

4.6 Experimental Design for the Determination of Model Parameters

In this thesis study some of the meta-heuristic algorithms are embedded into each other and new meta-heuristic algorithms are obtained. HGAQSA, GMASA and GASAVNS are the developed new meta-heuristic algorithms obtained with this technique. It can easily be concluded that the model parameters are correlated and affected by each other. As a result of this, it is difficult to guess the optimum or the suitable model parameter which will present the optimum solution in minimum number of schedule. In order to reveal the correlation between the parameters an experimental design is performed. Four meta-heuristic algorithms are examined in this analysis which are; GASA, HGAQSA, GMASA and GASAVNS. Although, GASA was not an algorithm developed in this thesis, it is included in the analysis. The reason of this is to measure the interaction between the basic parameters such as crossover, mutation and population size. The remaining algorithms require more parameters than GASA and it is not practical to examine effect of each parameter of those algorithms.

In experimental design analysis four algorithms are examined by using the 18, 29 and 63-activity projects. The projects are analyzed by considering the only overhead costs and delay penalty and early finish bonuses are not considered.

Experimental design is the systematic measurement of the responses of output variable based on the systematic changes on the input variables. The terminology of experimental design is briefly introduced. *Variable* is a qualitative or quantitative entity that can vary or take on different values. *Reliability* is a crucial characteristic of measurement and refers to the consistency of a measuring device. *Validity* of an instrument means that it measures what it is designed to measure. *Control* involves holding constant or varying variables systematically so that, their effects can be removed from a study or compared to other conditions. *Randomization* refers to the assignment of subjects to conditions or levels of an independent variable either by the investigator or by a natural process in the field (Lewis-Beck M. S. 1993).

The design of an experiment should take; the objectives of experiment, the number of factors under investigation, possible presence of identifiable and non-identifiable extraneous factors, amount of time and money available for the experimentation into account (Ryan 2007).

In this study, boundaries of input variables are determined by obtaining the most common numbers from the literature. After determining the minimum and maximum values of the variable, experimental design analysis is performed by spreadsheet method.

Main effect of a dependent variable on the independent variable is defined as the difference in the average response between the high and low levels of a factor.
The main effect can be represented as (Barrentine 1999);

$$E(A) = \overline{Y}_{A^+} - \overline{Y}_{A^-} \tag{4.1}$$

Where, $E(A)$ is the effect of dependent variable A on the independent variable, $\overline{Y}_{A^+}$ is the average response of the high level, $\overline{Y}_{A^-}$ is the average response of the low level of A.

Interaction occurs when a particular combination of two factors affect the dependent variable unexpectedly from simply observing their main effects. Interaction is defined as one-half of the difference between the effect of independent variable A at the high level of B and the effect of A at the low level of B. The interaction of dependent variable A and B can be formulated as (Barrentine 1999);

$$E(AB) = \frac{1}{2}\left[\left(\overline{Y}_{A^+} - \overline{Y}_{A^-}\right)_{B+} - \left(\overline{Y}_{A^+} - \overline{Y}_{A^-}\right)_{B-}\right] \tag{4.2}$$

where, $\left(\overline{Y}_{A^+} - \overline{Y}_{A^-}\right)_{B+}$ is the effect of A when B is high and $\left(\overline{Y}_{A^+} - \overline{Y}_{A^-}\right)_{B-}$ is the effect of A when B is low.

In order to determine the significance of the independent parameters and their interactions between each other, t-test is performed. Determination of significance requires calculation of standard deviation as a measure of inherent variation or experimental error in the process. Variance is the square of the deviation of each observation of a sample from the sample average which can be written as (Barrentine 1999);

$$S^2 = \frac{\sum (X_i - \overline{X})^2}{n-1} \tag{4.3}$$

Average variance is the average of the variance of each variance obtained by k runs, where k is equal to 2^n if there are n investigated independent variables with only high and low levels. Average variance is computed as following;

$$S_e^2 = \sum S_i^2 \tag{4.4}$$

Effects of the dependent variables are differences between averages and require definition of a modified variation which is called variation of the effects as (Barrentine 1999)

$$S_{eff}^2 = S_e^2 \frac{4}{N} \tag{4.5}$$

where N is the total number of trials. As long as the factors will have only high and low levels equation 4.5 will be valid.

In order to perform t-test, degrees of freedom of the data set should be determined. The computation of degrees of freedom is shown below (Berger and Maurer, 2002);

$$d.f. = (\#\ of\ \ observations\ \ per\ run - 1) \times (\#\ of\ \ runs) \tag{4.6}$$

Next step is selecting a significance level for the t-test. In this analysis 95% significance interval is preferred. By using the significance interval and degrees of freedom, t-value is obtained and decision limits are calculated by the formula (Berger and Maurer, 2002),

$$DL = \pm (t_{\alpha,df})(\sigma_{\alpha,df}) \tag{4.7}$$

If effect of a variable or interaction is outside the region defined by DL, then the variable or interaction is determined as significant. The model parameters are adjusted according to the significances of them. However, the relationships of the parameters are not always linear which makes interpolation not applicable.

4.6.1 Experimental Design of 18-Activity Project

18-activity project is analyzed for experimental design of GASA, HGAQSA, GMASA and GASAVNS. In order to limit computational burden and make any interpretation from the analysis easier, number of parameters for each algorithm to be examined is taken as 4. The project cost at the end of 50000th schedule is taken into account in order to make a fair comparison of the effect of the parameters.

First method to be analyzed is GASA. Variables to be examined and their low and high limits are given in Table 4-51.

Table 4-51 High and Low levels of parameters of GASA

Parameter	**High Level**	**Low Level**
Population Size (A)	200	50
Crossover (B)	0,9	0,3
Mutation (C)	0,9	0,3
Boltzmann Constant (D)	1,5	0,5

As there are four parameters number of interactions and parameters becomes 15.

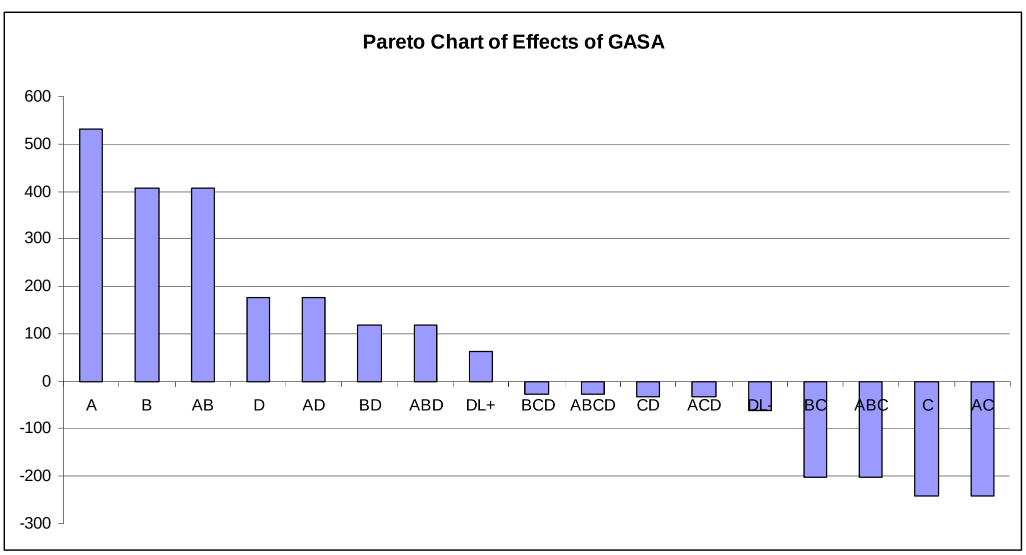

Figure 4.5 Pareto Chart of effects of GASA

In Figure 4.5 pareto chart of effects of GASA for the 18-activity project is shown. The bars show the effect of the parameter on the total cost of the project. The most significant parameter is the population size where if population size is increase total project cost at the end of the 50000th schedule also increase. Similarly, when crossover ratio and Boltzmann Constant is increased total project cost also increases. There is significant interaction between the parameters population size and crossover, population size and Boltzmann Constant and Crossover and Boltzmann Constant. As this is the case the interaction between the three parameters are also significant. The positive interaction means that when the population size and crossover rate is increased simultaneously, the increase in total project cost will be more than the prediction by only considering increase in total project cost when these two parameters are increased solely.

It is seen that increasing mutation rate decreases the total project cost at the end of the 50000th schedule. Consequently, in order to obtain near-optimum results at the end of the 50000th schedule low level values should be assigned to the population size, crossover and Boltzmann constant and high level value should be assigned to mutation.

The next meta-heuristic algorithm to be examined is HGAQSA. Examined parameters and their high and low level values are given in Table 4-52.

Table 4-52 High and Low levels of parameters of HGAQSA

Parameter	**High Level**	**Low Level**
Tunnel Strength Narrower (A)	0,99	0,7
QSA Period (B)	20	5
QSA iteration (C)	40	10
Boltzmann Constant (D)	1,0	0,5

The pareto chart of the effects of HGAQSA for the 18-activity project is given in Figure 4.6. From the analysis results it can be concluded that QSA period and Boltzmann Constant are positively correlated with the project cost obtained at the end of the 50000th schedule. On the other hand, QSA iteration is negatively correlated with the total project cost. It means that increasing the number of random walks per QSA iteration ends up with less cost projects. Tunnel Strength Narrower parameter do not have any significance in this project. In addition to this, when compared with other heuristics HGAQSA end up with very little variation at the end of the analysis that there are very little differences in the project costs obtained by different parameter combinations.

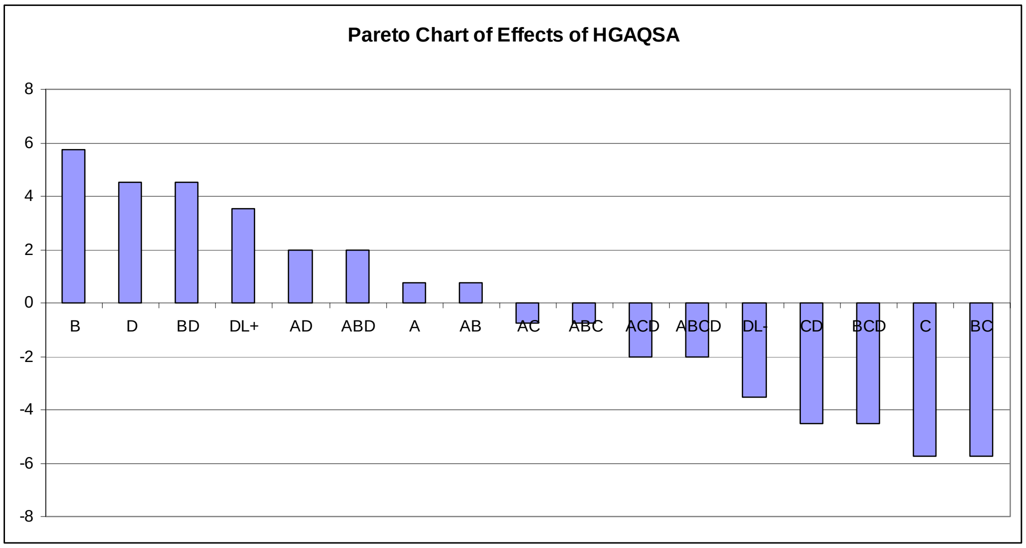

Figure 4.6 Pareto Chart of effects of HGAQSA

Most of the interactions between the parameters are insignificant which does not necessarily means that there is no interaction between the parameters. The 18-activity project was a too easy project which fails to reveal the interactions between the parameters.

After the analysis of HGAQSA, GMASA is analyzed. Parameters analyzed are shown in Table 4-53.

Table 4-53 High and Low levels of parameters of GMASA

Parameter	**High Level**	**Low Level**
Memetic Iteration (A)	3	1
Memetic Search Period (B)	20	5
Boltzmann Constant (C)	0,8	0,5
Population Size (D)	150	50

Pareto chart effects of GMASA are shown in Figure 4.7. It is obvious that population size and memetic search period has considerably high significance. In addition to this, number of memetic iteration is also significant. Increasing these parameters causes an increase in the total project cost at the end of the 50000th schedule. Interaction of memetic search period and population size is also significant. Especially population size should be decreased in order to obtain low cost schedules at the end of the 50000th schedule.

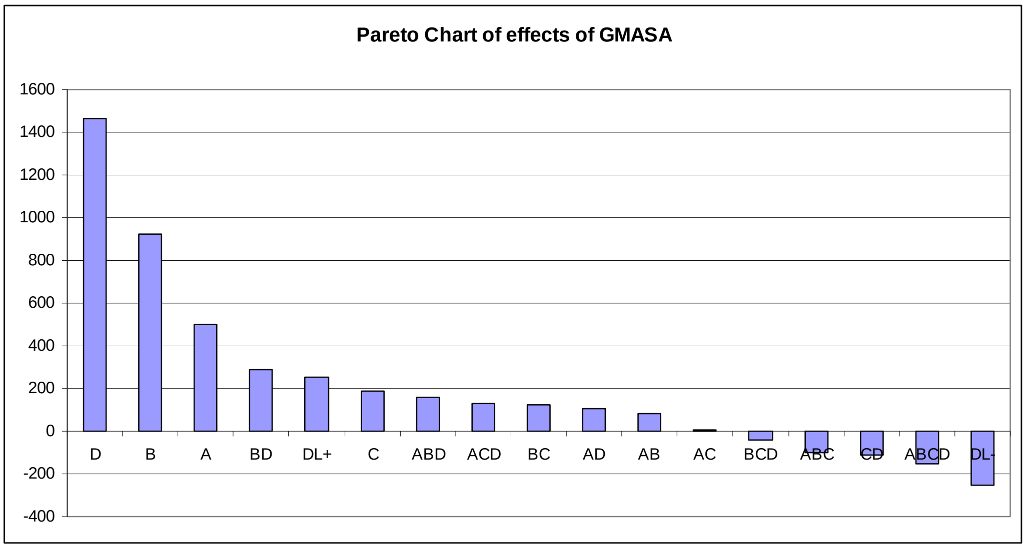

Figure 4.7 Pareto Chart of effects of GMASA

In the final analysis performed by 18-activity projects, GASAVNS meta-heuristic algorithm is analyzed. Parameters examined are shown in Table 4-54.

Table 4-54 High and Low levels of parameters of GASAVNS

Parameter	**High Level**	**Low Level**
VNS period (A)	20	5
VNS iteration (B)	20	5
Maximum Neighborhood (C)	5	2
Boltzmann Constant (D)	0,8	0,5

The pareto chart effects of GASAVNS is shown in Figure 4.8. The analysis results reveal that VNS iteration number, Maximum Neighborhood number and their interaction significantly affect the total project cost. The two parameters should be assigned low numbers in order to obtain low cost project schedule at the end of the 50000th schedule. On the other hand, VNS period is negatively correlated with the total project cost where increasing the VNS period ends up with less total project cost.

The significant parameters are the most important parameters of GASAVNS. Decreasing VNS iteration number decreases the number of random walks per VNS iteration. Similarly decreasing maximum neighborhood number, decreases maximum number of changes could be made in the crashing alternatives at once. On the other hand, increasing the VNS period decreases the number of VNS during the optimum search. Consequently, it can be concluded that VNS could not improve the convergence capability of GASA.

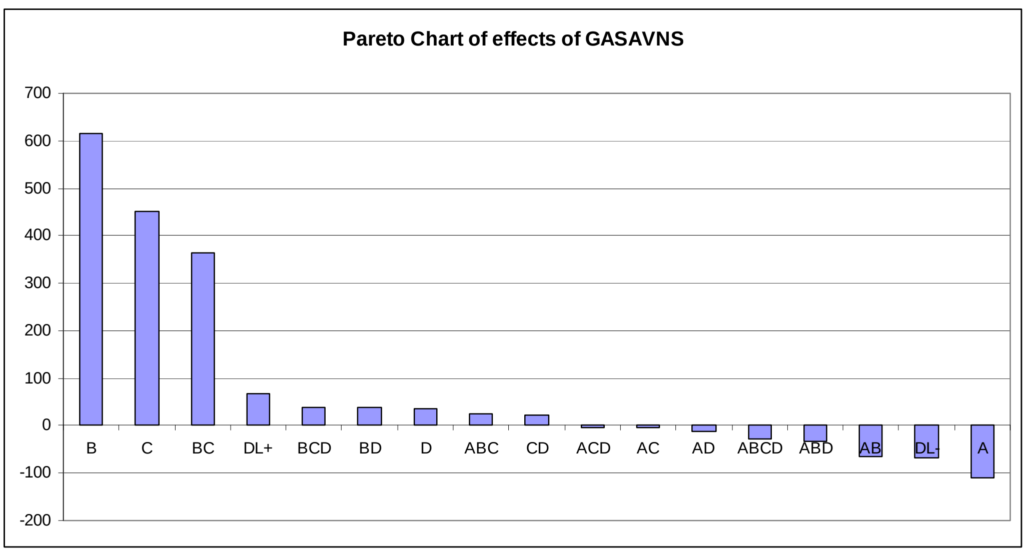

Figure 4.8 Pareto Chart of effects of GASAVNS

4.6.2 Experimental Design of 29-Activity Project

29-activity project is also examined by experimental design in order to monitor the characteristics and effects of the hybrid meta-heuristic algorithms. Similar to the previous analysis the total project cost obtained at the end of the 50000th schedule is taken into account. The objective function was to obtain the least cost project schedule for €1200/day constant overhead cost.

First meta-heuristic algorithm analyzed is GASA and the algorithms monitored parameters and their high level and low level values are given in Table 4-55.

Table 4-55 High and Low levels of parameters of GASA

Parameter	**High Level**	**Low Level**
Population Size (A)	200	50
Crossover (B)	0,9	0,3
Mutation (C)	0,9	0,3
Boltzmann Constant (D)	1,5	0,5

The high and low levels of the parameters are exactly the same with the 18-activity project. This will give some clue about how the algorithm reacts with different projects. The pareto chart of effects of GASA for the 29-activity project is given in Figure 4.9.

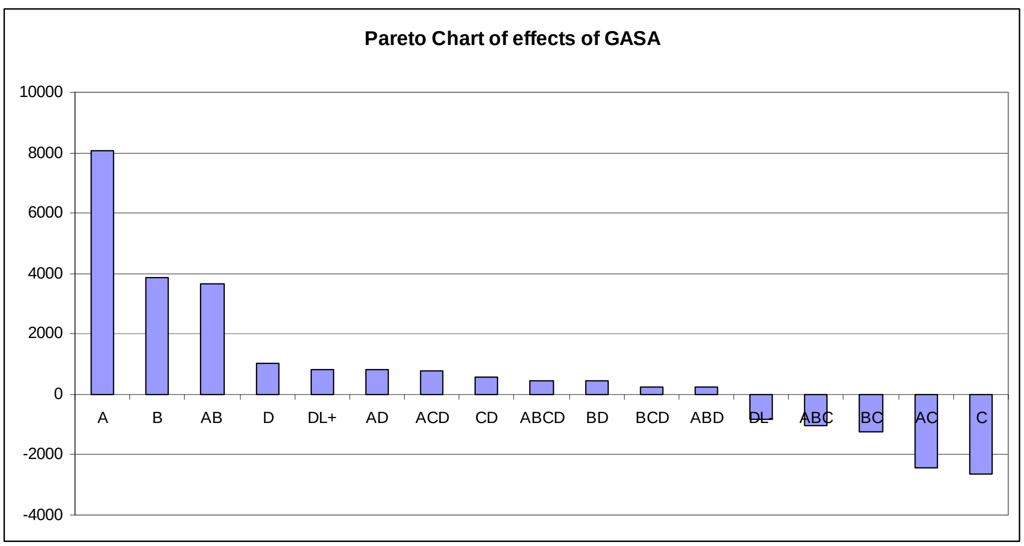

Figure 4.9 Pareto Chart of effects of GASA for 29-activity project

When the analysis results are examined it is seen that main effects are almost same with the 18-activity project. There are some minor changes on the magnitudes of the effects. It is seen that the effects of the parameters are about ten times higher than the previous analysis which means that the 29-activity project is more difficult than the 18-activity project.

It is seen that effect of population size is considerably higher than the other effects, which means large population size decreases the chance of obtaining cheaper project schedules at the end of the 50000th schedule. It is seen that some of the interactions becomes insignificant in this project. Interaction between crossover and BC, and some triple interactions can be given as example. The reason of this can be given as high effect of the population size. Some of the interactions have changed their sign, where interaction between ABCD and BCD can be given as example. However, both interactions are insignificant and therefore the sign change does not have a significant effect on the analysis results.

Second algorithm analyzed on 29-activity project was HGAQSA. Examined parameters and their low and high level values are given in Table 4-56.

Table 4-56 Analyzed HGAQSA parameters for the 29-activiy project

Parameter	**High Level**	**Low Level**
Tunnel Strength Narrower (A)	0,99	0,7
QSA Period (B)	20	5
QSA iteration (C)	40	10
Boltzmann Constant (D)	1,0	0,5

The analysis results are shown in Figure 4.10. Similar to GASA, almost same effects are obtained in the analysis performed on HGAQSA. However, in the 29-activity project HGAQSA could not obtained optimum results in all of the trials and as seen in Figure 4.10 the effects are higher. There are some minor changes in the interactions where the interaction between QSA Period and Boltzmann Constant, and interaction between QSA iteration and Boltzmann Constant are insignificant. The highest effect is measured on the parameter QSA iteration which means increasing the QSA period end up with high total project cost projects at the end of the 50000th schedules. This means that QSA random walks have significant effect on decreasing the total project cost and the more frequent QSA iterations are performed the better project schedules are obtained.

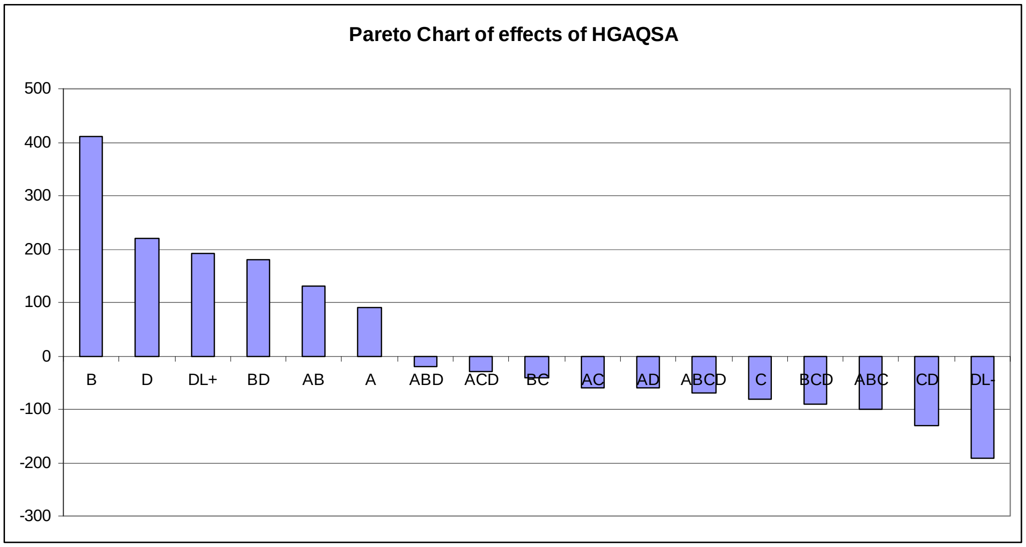

Figure 4.10 Pareto Chart of effects of HGAQSA for 29-activity project

Boltzmann Constant is also a significant parameter in which higher values of it end up with high project costs. The reason of it can be explained as high Boltzmann Constant values increases the probability of acceptance of a detrimental mutation. As this is the case, detrimental mutations prevent converging into global optimum and after a certain cooling convergence become possible. Tunnel strength narrower and QSA iteration number do not have significant effect.

Third algorithm analyzed on 29-activity project is GMASA. The parameters analyzed and high and low level values assigned are shown in Table 4-57.

Table 4-57 High and Low levels of parameters of GMASA

Parameter	High Level	Low Level
Memetic Iteration (A)	3	1
Memetic Search Period (B)	20	5
Boltzmann Constant (C)	0,8	0,5
Population Size (D)	150	50

Analysis results of GMASA are shown in Figure 4.11. Surprisingly all of the four parameters are significant. In this analysis Population Size has the most significant effect where higher population size end up with high total project cost at the end of the 50000th schedule. However the population size is not as significant as the on measured in the analysis performed on18-activity project.

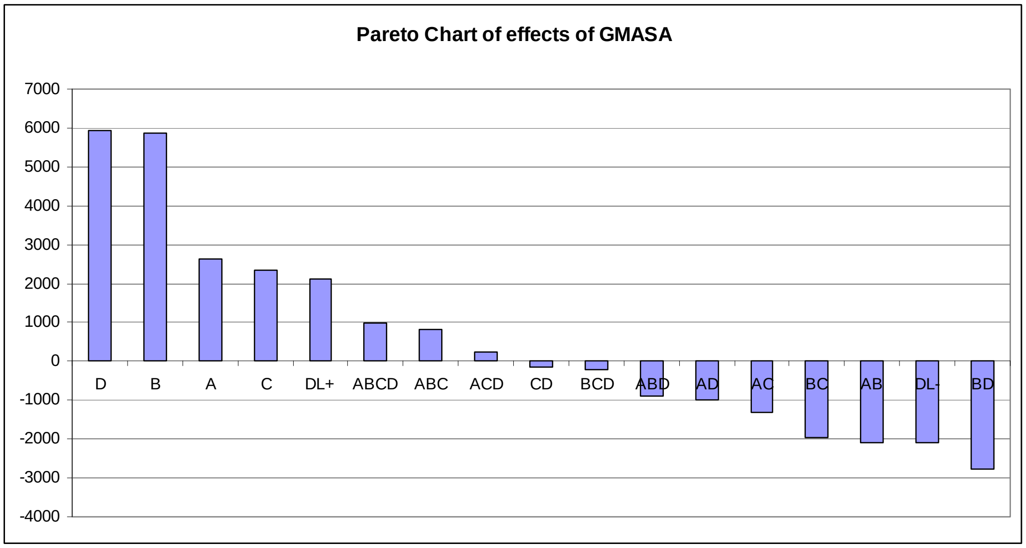

Figure 4.11 Pareto Chart of effects of GMASA

Memetic search period, Memetic iteration number and Boltzmann constant have also significant effect in which if high values are assigned to these parameters, high total project costs are expected at the end of the 50000th schedule. High number of memetic search slows down the convergence ration, however memetic search is beneficial that frequently performing memetic search increases convergence ratio. However, too high number of memetic search slows down the convergence speed. In addition to this, low values should be assigned to Boltzmann Constant in order to prevent acceptance of detrimental mutations.

Fourth meta-heuristic algorithm analyzed is GASAVNS. Parameters examined and the high level and low level values of the parameters are shown in Table 4-58.

Table 4-58 High and Low levels of parameters of GASAVNS

Parameter	**High Level**	**Low Level**
VNS period (A)	20	5
VNS iteration (B)	20	5
Maximum Neighborhood (C)	5	2
Boltzmann Constant (D)	0,8	0,5

Analysis results are shown in Figure 4.12. The results are very similar to the results of the 18-activity project. In this case, number of VNS iterations becomes more significant. The interaction between maximum neighborhood and number VNS iteration is still significant. Both analysis results conducted on 18-activity and 29-activity projects concludes that VNS does not improve the performance of GASA.

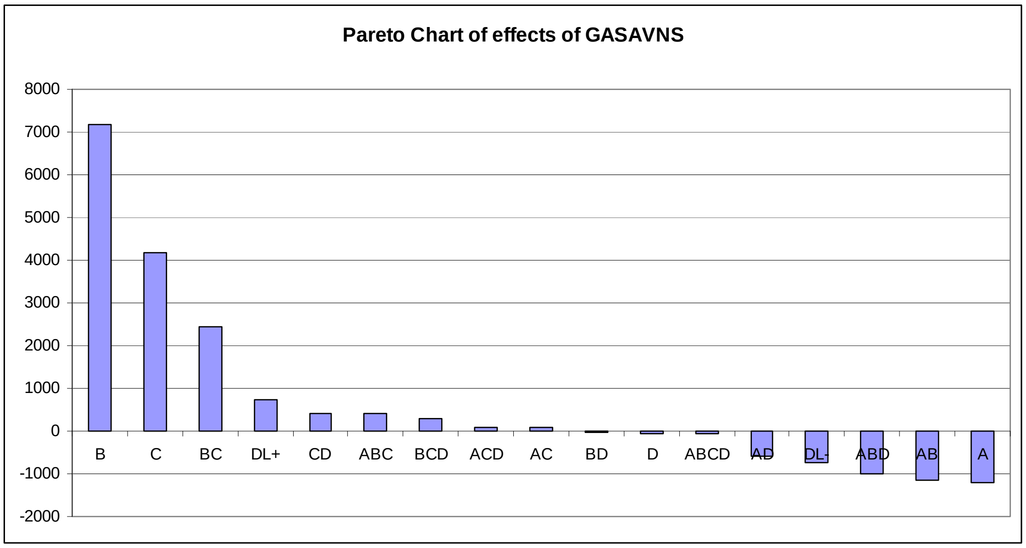

Figure 4.12 Pareto Chart of effects of GASAVNS for 29-activity project

4.6.3 Experimental Design of 63-Activity Project

The 18-activity and 29-activity projects were relatively small projects with limited number of local minima. In order to perform the experimental design analysis on a relatively challenging network, 63-activity project is also analyzed. Since the 63-activity project is more difficult, analysis are stopped at the end of the 250000th schedule for this analysis. The effect of the meta-heuristic algorithms parameters are investigated based on the total project cost at the end of the 250000th schedule. The project is analyzed with $2300/day constant overhead cost without any delay penalty or early finish bonus.

GASA is the first algorithm analyzed. The model parameters with the high level and low level values are given in Table 4-59.

Table 4-59 High and Low levels of parameters of GASA

Parameter	**High Level**	**Low Level**
Population Size (A)	250	100
Crossover (B)	0,8	0,3
Mutation (C)	0,9	0,4
Boltzmann Constant (D)	1,5	0,5

High level and low level values are slightly adjusted based on the analysis results of the 18 and 29-activity projects. Although increasing population size has detrimental effect on the total project cost, population size is slightly increased as number of the parameters to solve is almost tripled when compared with the previous projects.

Analysis results of GASA are given in Figure 4.13. Effects of GASA for the 63-activity project are significantly different than the effects for the 29 and 18-activity projects. The project size and the difficulty of the project have high effect on this result.

First of all population size has significant negative effect which was the reverse in the previous analysis. Crossover has the most significant effect and higher the crossover ratio higher the total project cost. Although, high level value of crossover is decreased by 0,1 high level value of crossover ratio decreases the convergence speed. Surprisingly, increasing mutation ratio decreases the convergence ratio which also contradicts with the previous analysis results. It can be inferred from the experimental design results that for small sized projects high mutation ratio is better. For medium and large sized projects, low mutation ratio increases the convergence speed. Also the analysis results mentioned that, population size should be increased when solving large projects.

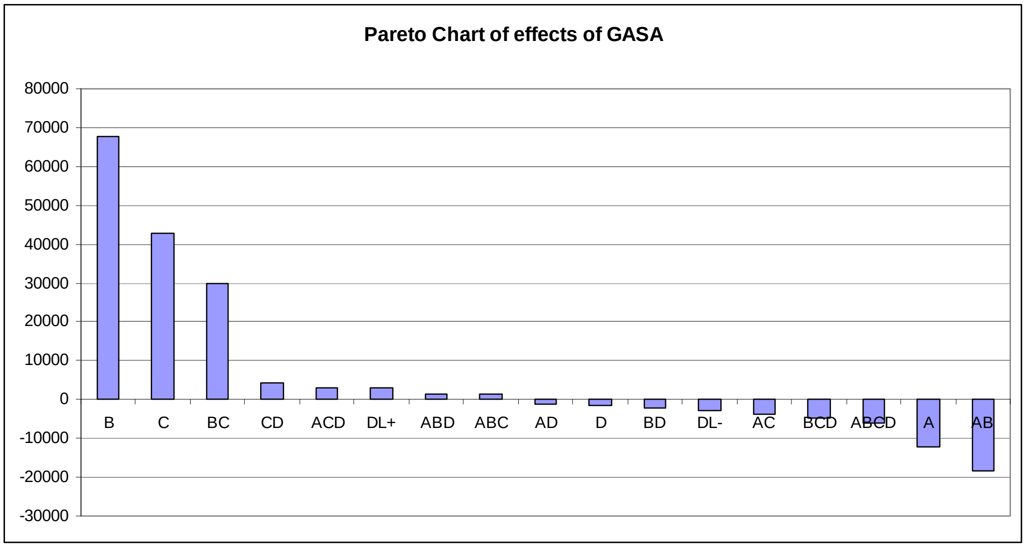

Figure 4.13 Pareto Chart of effects of GASA for 63-activity project

HGAQSA is the second meta-heuristic algorithm which is analyzed by the 63-activity project. Parameters analyzed and their high level and low level values are given in Table 4-60.

Table 4-60 High and Low levels of parameters of HGAQSA

Parameter	**High Level**	**Low Level**
Tunnel Strength Narrower (A)	0,99	0,7
QSA Period (B)	20	5
QSA iteration (C)	100	25
Boltzmann Constant (D)	1,0	0,5

Analysis results of HGAQSA are shown in Figure 4.14. The analysis results of HGAQSA are also different when compared with the 18 and 29-activity projects' analysis results. In this case there is only one significant parameter which is QSA period. The analysis results courages more frequent QSA iterations which is the same with the previous cases. However, in the 63-activity project increasing Boltzmann Constant would present less total project cost. On the other hand, this result is not significant but a controversy when compared with the previous analysis.

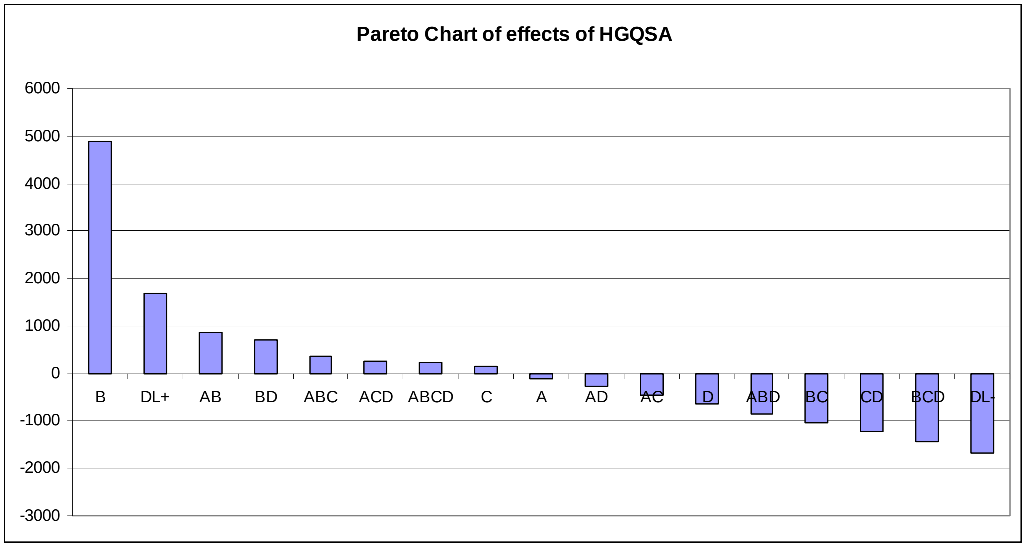

Figure 4.14 Pareto Chart of effects of HGAQSA

GMASA is the third algorithm analyzed on 63-activity project. Parameters analyzed and the high and low level values of the parameters are shown in Table 4-61.

Table 4-61 High and Low levels of parameters of GMASA

Parameter	High Level	Low Level
Memetic Iteration (A)	3	1
Memetic Search Period (B)	20	5
Boltzmann Constant (C)	0,8	0,5
Population Size (D)	250	100

The population size is slightly increased and the remaining parameters kept constant in this analysis. The analysis results are shown in Figure 4.15. Effects of parameters are very different than the previous analysis. In this case Memetic Iteration has the most significant effect on the results. Assigning high values for the memetic iteration number significantly decreases the convergence speed. In addition to this, large population size also decreases the convergence speed. This is an unexpected result for the 63-activity project. The analysis results reveal that frequent memetic search decreases the convergence speed. When the effects are covered altogether it is seen that memetic search algorithm slows down the convergence speed of GASA when the 250000th schedule is considered.

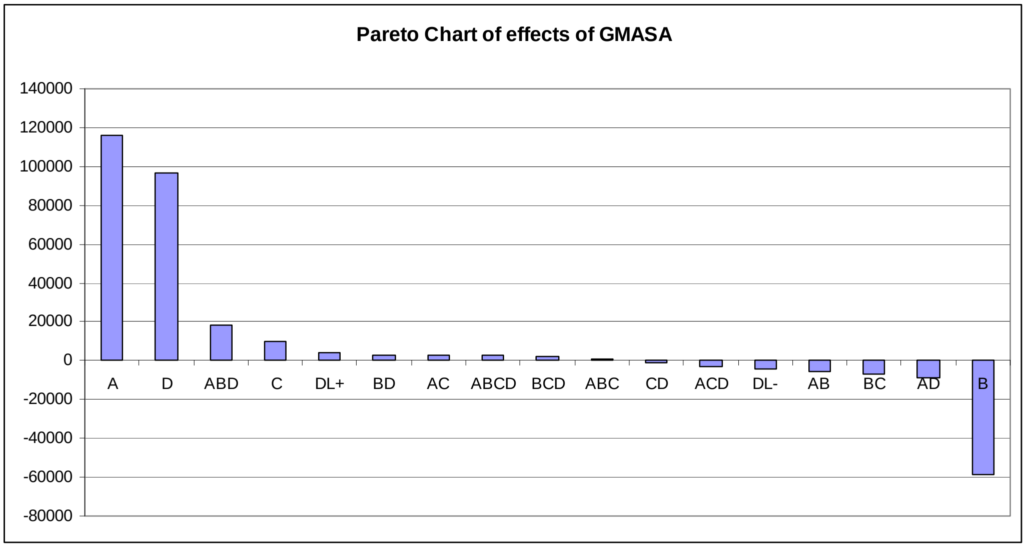

Figure 4.15 Pareto Chart of effects of GMASA

Fourth meta-heuristic algorithm which is analyzed by 63-activity project is GASAVNS. The parameters analyzed and their high level and low level values are given in Table 4-62.

Table 4-62 High and Low levels of parameters of GASAVNS

Parameter	**High Level**	**Low Level**
VNS period (A)	20	5
VNS iteration (B)	20	5
Maximum Neighborhood (C)	5	2
Boltzmann Constant (D)	0,8	0,5

Analysis results of GASAVNS are shown in Figure 4.16. There is not a notable change in the effects when compared with the previous analysis of GASAVNS. Number of VNS is still the most significant parameter and assigning high number of VNS search slows down the convergence speed at the end of the 250000th schedule. In addition to this, increasing the neighborhood and frequent VNS search slow down the convergence speed. It can be concluded that VNS does not improve the convergence capability of GASA when the project costs at the end of the 50000th and 250000th schedules are considered. On the other hand, GASAVNS is able to obtain global optimum when the maximum schedule number is increased.

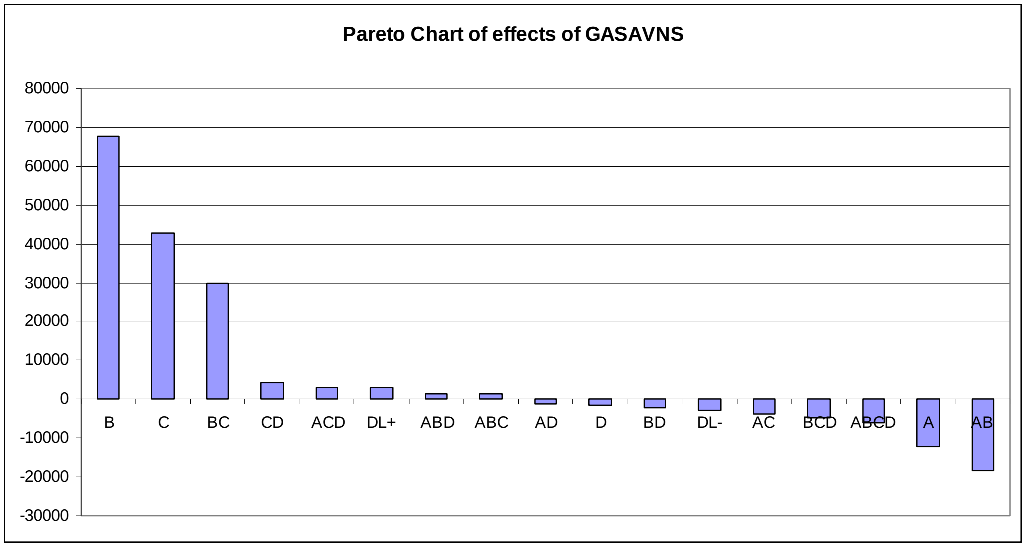

Figure 4.16 Pareto Chart of effects of GASAVNS for the 63-activity project

4.7 Convergence characteristics of the meta-heuristic algorithms

In this subchapter, convergence characteristics of the meta-heuristic algorithms are examined using the 18-activity project. The improvements in the convergence on GA based meta-heuristic algorithms by hybridizing and improvements on ACO, ESS and ESS by randomly shuffling the population at the end of a certain iteration number is analyzed. The additional computation duration caused and the gained improvement is also considered.

The first analyzed meta-heuristic algorithm is GA. Best individual's cost and population mean is shown in Figure 4.17. It is seen that because of mutation and natural selection operators the best value can be terminated. Improvement of the best individual is limited with $10.000 which is not a significant improvement. In addition to this, population mean fluctuates in a short bandwidth and does not improve as the iteration progresses. As a result of this, the probability of obtaining better genes from crossover and mutation does not increase since the gene quality of the population does not improve.

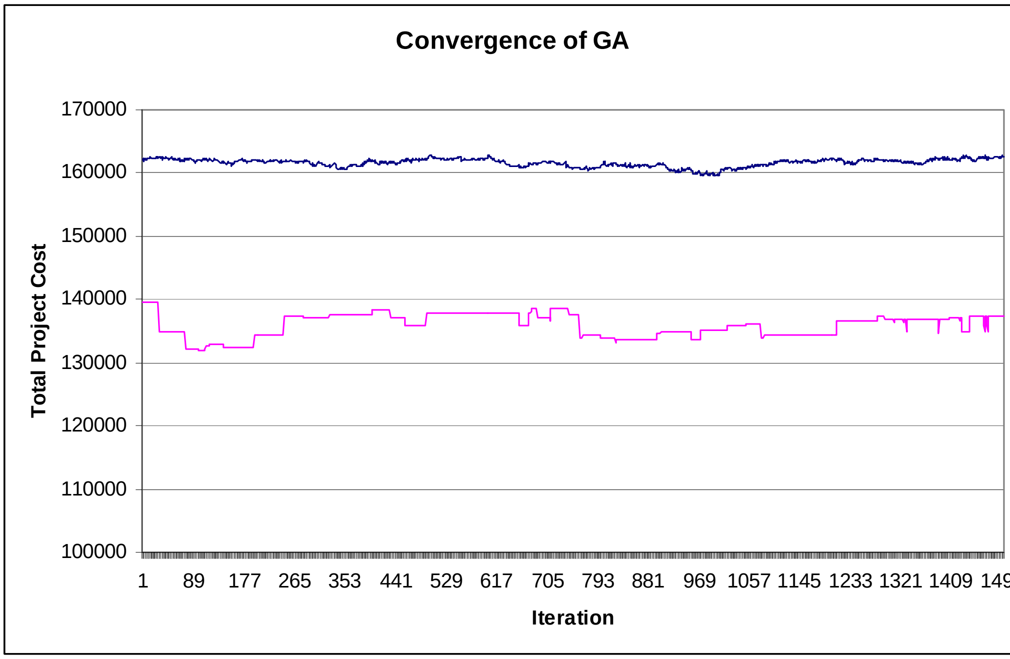

Figure 4.17 Convergence graph of GA.

In Figure 4.18 convergence graph of GASA is shown. SA significantly improves the gene quality when compared with the convergence of GA. Both algorithms start with almost same population mean but there is significant difference when the stopping criterion is met. Population mean of GASA decreases step by step, when iteration progresses and even catches global optimum.

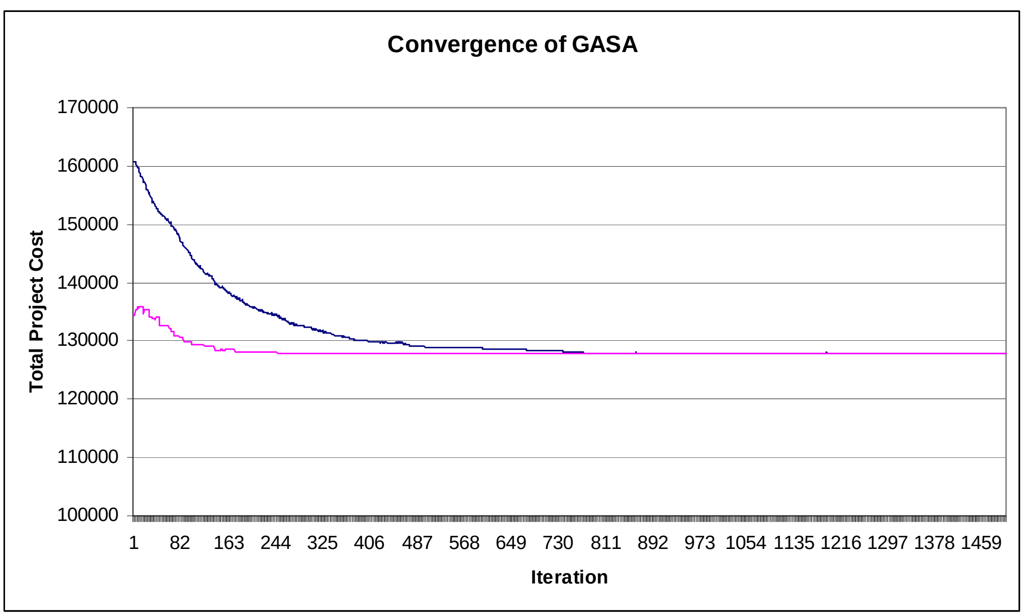

Figure 4.18 Convergence graph of GASA.

Improvement of population mean, increases gene quality and possibility of obtaining proper gees by crossover and mutation increases.

In Figure 4.19 convergence graph of HGAQSA is shown. Ladder type of convergence is the characteristics of HGAQSA. Sharp decreases both in population mean and best value mean that the local random walk session had been executed. Local search increases the convergence speed. In GASA, global optimum is obtained when the population mean value is significantly reduced. However, in HGAQSA global optimum is obtained when the difference between mean and best value is around \$10.000. This indicates that local search helps obtaining global optima without requiring very high quality population genes.

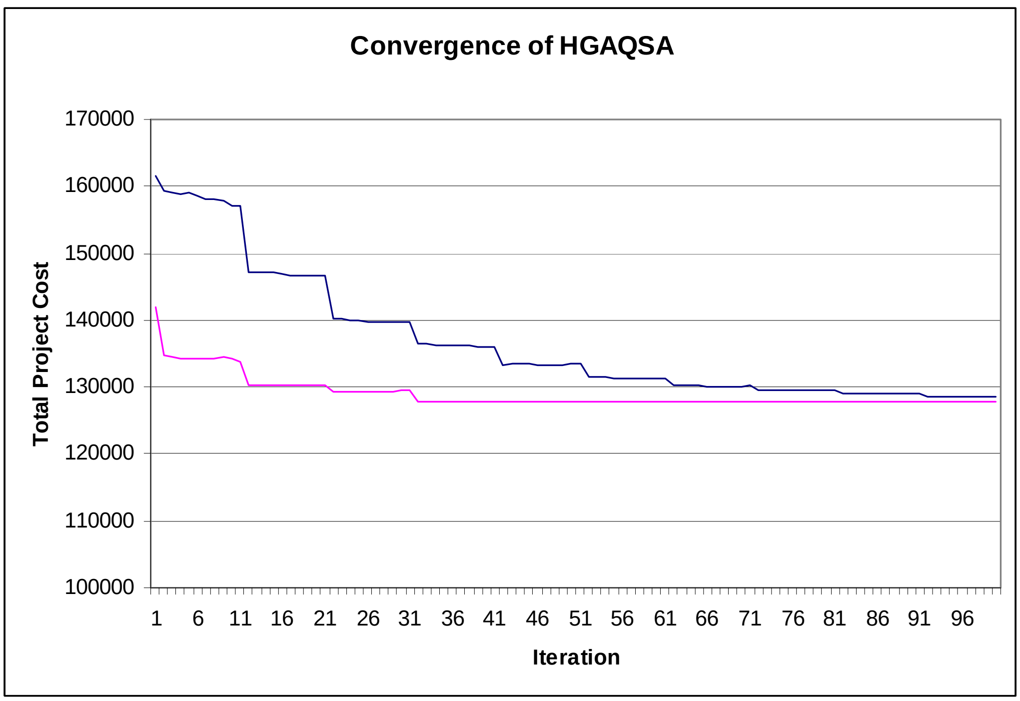

Figure 4.19 Convergence graph of HGAQSA

In Figure 4.20 convergence graph of GMASA is shown. Local memetic search is executed at the end each 10th iteration. The MA improves the population mean but it could not improve the population best. However, as the gene quality improves probability of producing high quality individuals by crossover and mutation operator increases. As a result of this, population best is improved after a number of GASA iterations.

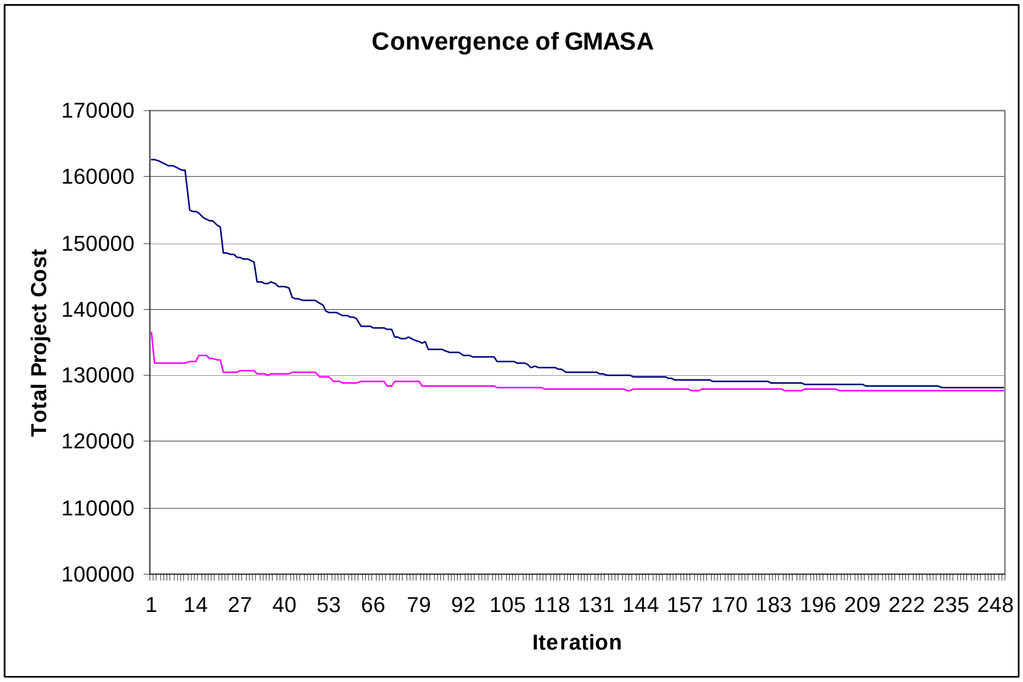

Figure 4.20 Convergence graph of GMASA

In Figure 4.21 convergence graph of GASAVNS is shown. The convergence characteristics of GASAVNS are very similar to the GMASA. Both methods obtain the global optima in more or less same local search.

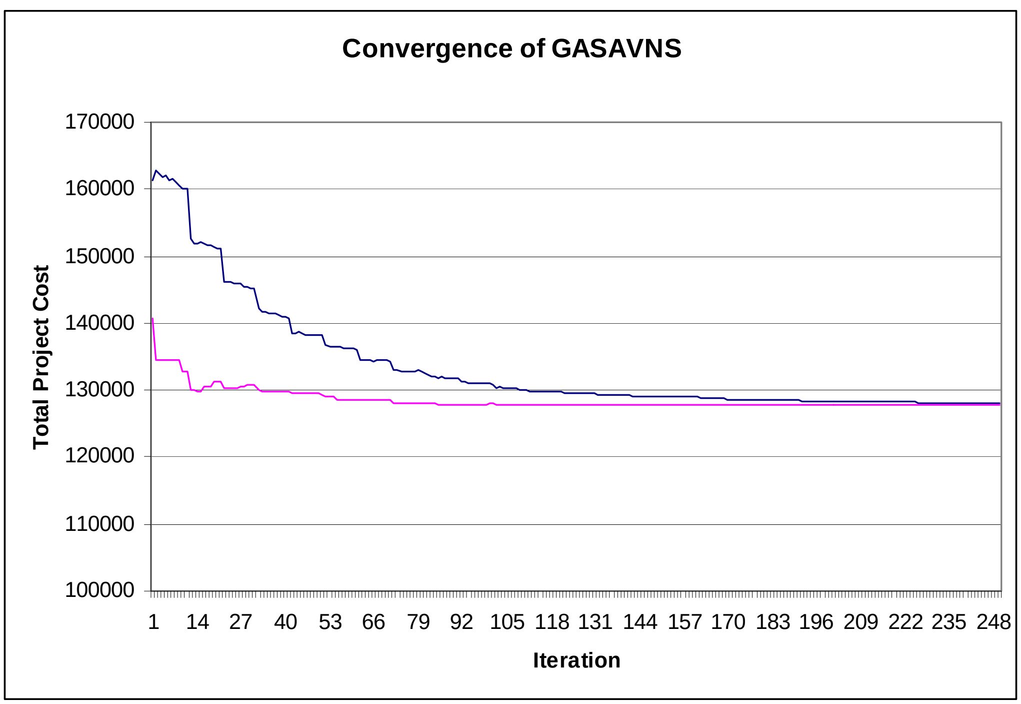

Figure 4.21 Convergence graph of GASAVNS

In Figure 4.22 convergence of PSO is shown. One full cycle of PSO ends at the end of the 50th iteration and the population is shuffled randomly and again PSO meta-heuristic algorithm starts. From the convergence graph, point of random shuffling can easily be seen. Population mean increases at the end of random shuffling, however after some PSO iterations an improvement in population best is obtained. Random shuffling helps PSO to escape from local optima and helps obtaining global optimum.

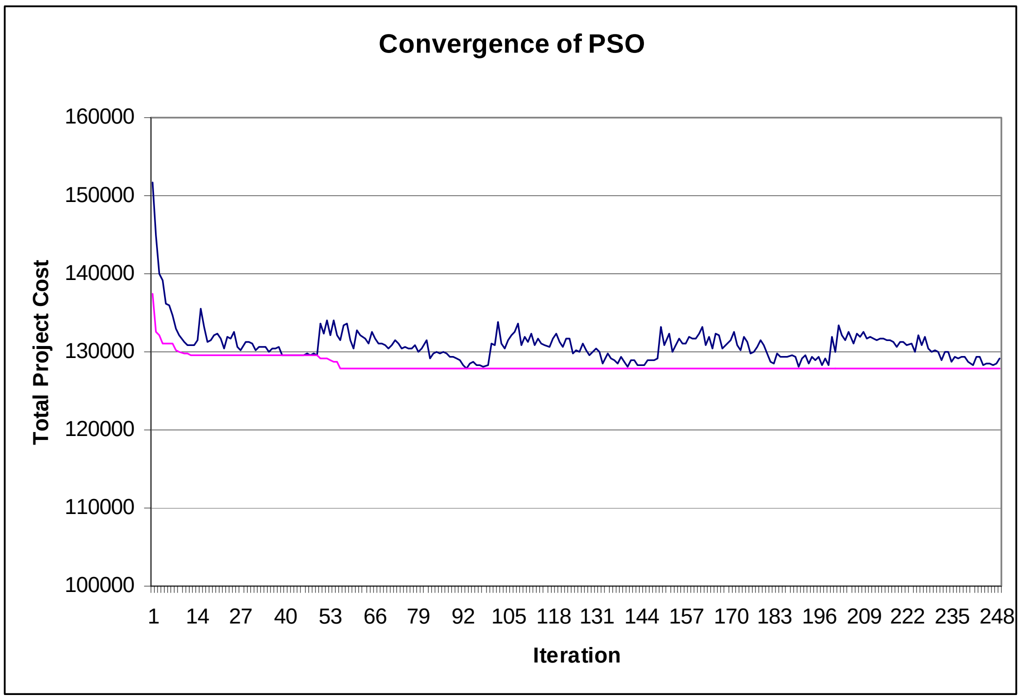

Figure 4.22 Convergence graph of PSO

In Figure 4.23 convergence graph of ACO is shown. Although ACO is shuffled randomly after certain iterations of ACO, there is not a significant mark on the population mean and best values which deviates because of random shuffling. Only the random shuffling performed at the 300th iteration help to decrease the total project cost. However, each random iteration increases the probability of escaping from the local minima.

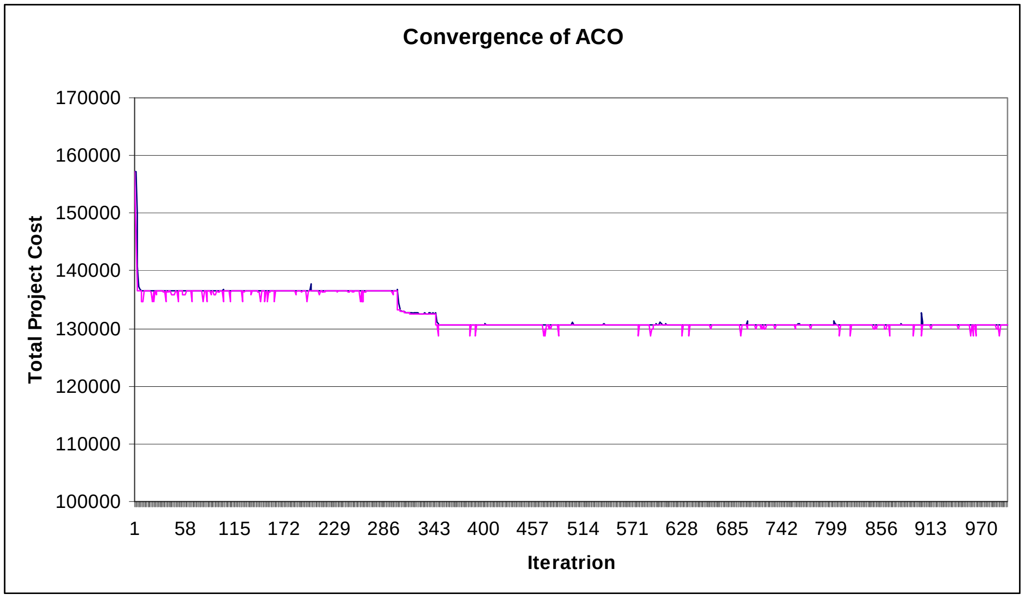

Figure 4.23 Convergence graph of ACO

In Figure 4.24 convergence graph of ESS is shown. It is seen that random shuffling of population has detrimental effect on the population mean. However, slight improvements are obtained in the global best up to 40th random shuffling. It is seen that, random shuffling significantly increases the computational demand and provides slight improvement in the analysis results.

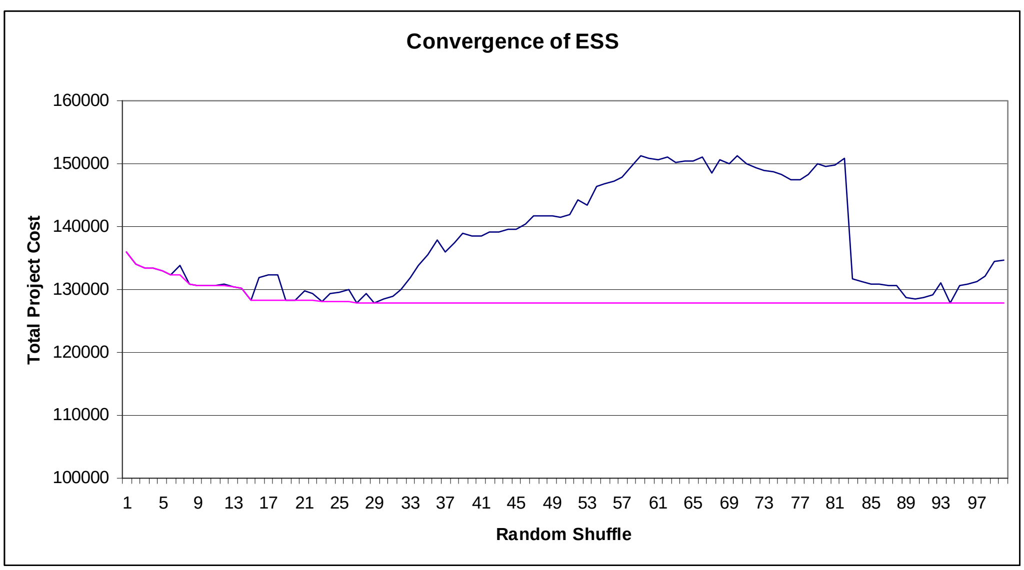

Figure 4.24 Convergence graph of ESS

CHAPTER 5

RESOURCE LEVELING

In this chapter, resource leveling of some construction projects found in the literature is performed by five GA based meta-heuristic algorithms which are GA, GASA, HGAQSA, GMASA and GASAVNS. The reason of this is the suitability of GA meta-heuristics for the resource leveling. Float of an activity also depends on the delayed amount of its predecessors. Because of this, representation of delay duration should be suitable for the changes of the delay periods of its predecessors. For this reason, representation style is adapted for the delay representation of resource leveling problems.

The gene representation of the meta-heuristic algorithms which is adopted for the resource leveling are explained in the next subchapter. Then construction projects analyzed are briefly introduced and the analysis results are given.

5.1 Gene representation

The resource leveling aims to minimize the fluctuations and peaks of the resource usage during the construction period. Resource leveling assumes unlimited resources so that delay of project completion time obtained by CPM scheduling is not allowed. As a result of this, only the activities which are not on the critical path can be delayed.

Resource leveling algorithms aims to minimize the sum of the squares of the cost functions of the daily resource usages. The cost function can be shown in Eq 5.1 (Leu et al. 2000);

$$\min \quad \sum_{i=0}^{r}\sum_{j=0}^{T} d_{ij}^{2} w_i \qquad (5.1)$$

Where r is the number of resources used in the project and d_{ij} is the daily resource usage of the resource i in the day j, w_i is the weight of the resource and T is the project duration. Weights of the resources can be assigned as 1 or the daily unit cost of that resource.

In this study, resource leveling is performed on the basis of delaying the activities within their total floats without any priority rule. Genes represent the information necessary for computing the delay of that activity. Similar to the TCT analysis binary representation is preferred in resource leveling. In contrast with the TCT analysis, only the activities which have slack time are represented by genes. As a result the gene length of an individual becomes the number of activities which have floats, times the bit per activity.

The method applied for computing the delay of an activity is very similar to the method for determining the crashing alternative. The gene representing the delay time of an activity covers a wide range. This range is the exponential of 2 since the binary representation is preferred. If 6 bit per activity is assumed the range length will be 64, if 8 bit per activity is assumed the range length will be 256.

The critical activities are not represented in the gene of the individual since the critical activities can not be delayed. As a result of this, number of parameters is reduced and search space is narrowed.

In the beginning, the project is scheduled by forward and backward passes and early start (ES) and late start (LS) times of the activities are determined. If ES of an activity equal to its LS than the activity is assigned as critical activity and its delay information is not represented by the genes.

Population is generated by assigning 0s and 1s randomly to the genes. Project is scheduled according to its gene, start time of activity is labeled as delayed start (DS) and finish time of an activity is labeled as delayed finish (DF). Delayed durations of the activities are determined by considering the logical relationships between its predecessors and the delay amount which is represented on its genes. Scheduling is performed by only forward pass, since every activities' LS time are pre-known and their total float can be computed by counting the working days between LS and its earliest start time obtained from the scheduling. The total float of that activity will be the maximum delay time of that activity. The delay time is computed as follows;

$$DT_i = TF_i * \frac{G_i}{G} \tag{5.2}$$

where, TF_i is the total float of the i^{th} activity, G_i is the encoded gene value of the i^{th} activity and G is the range. With this scheduling method it is guaranteed that the critical path will not be delayed. Although gene representing the activity has not changed, delay of that activity may change. This can happen by the changes of the DS of the predecessors of the activity. If the predecessors have not delayed, the activity can be shifted as much as the activity's initial TF. However, if the predecessors are delayed as much as they can, then the activity becomes critical and it can not be delayed. As a result, delay of an activity is determined by considering its gene and the floats of its predecessors.

After the scheduling of the project, daily resource usages are computed and the resource profile is evaluated. There is not any difference in crossover, mutation and natural selection operators. Seven projects obtained from the literature are analyzed by the five meta-heuristic algorithms.

5.2 Resource Leveling Analysis

In this sub-chapter the construction projects are briefly introduced and the obtained results are commented. The first two projects are taken from Son and Skibniewski (1999). CPM logical relationships and resource demand of the first project is given in Table 5-1.

The best value for the first analysis obtained by Son and Skibniewski (1999) is 6225. This value is obtained by the four of the meta-heuristic algorithms in all of the 10 trials. GASA required slightly more than 150.000 scheduling in order to converge into the best optimum value obtained so far. The remaining three meta-heuristic algorithms, especially HGASA, converged into global optimum by requiring very little number of iteration. Unfortunately GA could not obtained global optimum, although the algorithm performed the highest number of scheduling. The schedule which gives the least fluctuation of resources is given in Table 5-2.

Table 5-1 Project-1 obtained from Son and Skibniewski (1999)

Activity ID	**Duration**	**Predecessor**	**Resource Demand**
A	5	NULL	6
B	10	NULL	3
C	10	A	5
D	5	B	4
E	5	B	6
F	10	C	4
G	5	C	7
H	10	D	0
I	5	E	5
J	10	E	6
K	5	F	8
L	10	G, H, I	8
M	5	K, L, J	9

Table 5-2 Schedule of project-1 from Skibniewski and Son (1999)

Activity	**Start**	**Finish**	**Dur**
A	0	5	5
B	0	10	10
C	5	15	10
D	10	15	5
E	10	15	5
F	20	30	10
G	15	20	5
H	15	25	10
I	20	25	5
J	15	25	10
K	30	35	5
L	25	35	10
M	35	40	5

Resource usage of the first project of Son and Skibniewski (1999) is given in Table 5-3.

Table 5-3 Daily resource usage of project 1

Day	Resource Usage	Square of R.U.
1	9	81
2	9	81
3	9	81
4	9	81
5	9	81
6	8	64
7	8	64
8	8	64
9	8	64
10	8	64
11	15	225
12	15	225
13	15	225
14	15	225
15	15	225
16	13	169
17	13	169
18	13	169
19	13	169
20	13	169
21	15	225
22	15	225
23	15	225
24	15	225
25	15	225
26	12	144
27	12	144
28	12	144
29	12	144
30	12	144
31	16	256
32	16	256
33	16	256
34	16	256
35	16	256
36	9	81
37	9	81
38	9	81
39	9	81
40	9	81
Sum	485	6225

Analysis results of the first project of Son and Skibniewski (1999) are given in Table 5-7. The table consists of ten columns in which results of five meta-heuristic algorithms are illustrated. First column represents the best evaluation value obtained by the algorithm and the second column represents the total schedule evaluation. The analyses are repeated 10 times in order to examine the deviation of the results.

Table 5-4 Project-2 obtained from Son and Skibniewski (1999)

Activity ID	**Duration**	**Predeccessor**	**Resource Demand**
A	8	NULL	2
B	3	A	3
C	5	B, E	3
D	3	I	4
E	3	D	2
F	3	D	3
G	4	F, K	4
H	3	G	4
I	6	NULL	3
J	5	I	3
K	5	J	3

Second test problem is given in Table 5-4. Same result is also obtained in the analysis of the second project of Son and Skibniewski. In their study, the best value obtained is announced as 915. The schedule which gives the best objective value is given in Table 5-5.

Table 5-5 Schedule of second project of Son and Skibniewski (1999)

Activity	**Start**	**Finish**	**Duration**
A	0	8	8
B	13	16	3
C	18	23	5
D	6	9	3
E	11	14	3
F	9	12	3
G	16	20	4
H	20	23	3
I	0	6	6
J	8	13	5
K	13	18	5

Daily resource usage of the project-2 obtained from Son and Skibniewski is given in Table 5-6.

Table 5-6 Daily resource usage of project 2

Day	**Resource Usage**	**Square of R.U.**
1	5	25
2	5	25
3	5	25
4	5	25
5	5	25
6	5	25
7	6	36
8	6	36
9	7	49
10	6	36
11	8	64
12	8	64
13	5	25
14	6	36
15	6	36
16	6	36
17	7	49
18	7	49
19	7	49
20	7	49
21	7	49
22	7	49
23	7	49
Sum	143	915

Other test problems are obtained from Hiyassat (2000), Leu et al. (2000), Lu and Lam (2008) and Woodworth and Willie (1975). From Leu et al. two test problems are obtained. One of the test problems of Leu et al. contains three resource types and the remaining test problems contain only one resource.

The multi-resource test problem of Leu et al. is solved by assigning same weights to all resource types. The test problem showed that the algorithms are capable of solving multi-resource resource leveling problems.

Analysis result of the test problem analyzed by Hiyassat (2000) is given in Table 5-9. All of the meta-heuristic algorithms obtained the same schedule with the one obtained by Hiyassat. For the first test problem of Leu et al. (2000); GASA, HGASA, GMASA and GASAVNS obtained the same results with the Leu et al.. GA obtained same results in only seven trials out of 10. Analysis results of this problem are given in Table 5-10.

Second test problem of Leu et al. which contains three resource types is analyzed and five meta-heuristic algorithms obtained same schedule with the schedule obtained by Leu et al.. Analysis results of the second test problem of Leu et al. are given in Table 5-11.

Test problem of Lu and Lam (2008) and Woodworth and Willie (1975) are also analyzed and same results are obtained in all of the ten trails with the five meta-heuristic algorithms. Analysis results of Lu and Lam (2008) is given in Table 5-12 and results of Woodworth and Willie (1975) is given in Table 5-13.

Test problems and the obtained schedules are not given in order to limit number of tables if same results are obtained with the previous studies. Test problems and the schedules can be obtained from the referenced journals.

Table 5-7 Analysis results of Son and Skibniewski (1999) of project-1

	GA		GASA		HGASA		GMASA		GASAVNS	
	Obj. Func.	**P. Eval**	**Obj. Func.**	**P. Eval**	**Obj. Func.**	**P. Eval**	**Obj. Func.**	**P. Eval**	**Obj. Func.**	**P. Eval**
1	6461	301763	6225	158765	6225	22592	6225	57259	6225	41891
2	6357	301387	6225	158275	6225	22482	6225	58284	6225	42906
3	6373	301806	6225	158949	6225	22621	6225	58658	6225	42343
4	6325	301971	6225	158868	6225	22652	6225	59049	6225	41892
5	6361	301800	6225	158185	6225	22701	6225	57969	6225	42779
6	6361	301868	6225	158699	6225	22535	6225	57803	6225	43109
7	6369	301868	6225	158760	6225	22679	6225	59273	6225	42216
8	6381	301749	6225	158616	6225	22716	6225	57805	6225	43253
9	6357	301739	6225	158378	6225	22777	6225	58868	6225	42912
10	6279	301476	6225	158947	6225	22607	6225	56892	6225	43077

Table 5-8 Analysis results of Son and Skibniewski (1999) of project-2

	GA		GASA		HGASA		GMASA		GASAVNS	
	Obj. Func.	**P. Eval**	**Obj. Func.**	**P. Eval**	**Obj. Func.**	**P. Eval**	**Obj. Func.**	**P. Eval**	**Obj. Func.**	**P. Eval**
1	923	301989	915	158636	915	35665	915	56612	915	41070
2	941	301513	915	158728	915	35667	915	56842	915	41477
3	937	301325	915	158545	915	35909	915	57109	915	41157
4	935	301709	915	158237	915	35785	915	56347	915	41099
5	939	301830	915	158823	915	35758	915	56715	915	42299
6	935	301816	915	158998	915	35775	915	56326	915	41062
7	923	301563	915	158973	915	35822	915	56450	915	41097
8	939	302303	915	158551	915	35647	915	56525	915	41757
9	939	301115	915	159087	915	35829	915	57190	915	41204
10	935	301503	915	158667	915	35846	915	55481	915	40995

Table 5-9 Analysis results of the project Hiyassat (2000)

	GA		GASA		HGASA		GMASA		GASAVNS	
	Obj. Func.	**P. Eval**	**Obj. Func.**	**P. Eval**	**Obj. Func.**	**P. Eval**	**Obj. Func.**	**P. Eval**	**Obj. Func.**	**P. Eval**
1	2704	301436	2704	159049	2704	36288	2704	57525	2704	41447
2	2704	301843	2704	158265	2704	36495	2704	57636	2704	40869
3	2704	302757	2704	158144	2704	36289	2704	57725	2704	41243
4	2704	302247	2704	158939	2704	36309	2704	57143	2704	41209
5	2704	301998	2704	157774	2704	36488	2704	57973	2704	41578
6	2704	302087	2704	159418	2704	36209	2704	57276	2704	41035
7	2704	302181	2704	158361	2704	36217	2704	57027	2704	41133
8	2704	301750	2704	158425	2704	36417	2704	58174	2704	41855
9	2704	302249	2704	158881	2704	36451	2704	58199	2704	41753
10	2704	301636	2704	158694	2704	36306	2704	57287	2704	42220

Table 5-10 Analysis results of Leu et al. (2000) for the first project

	GA		GASA		HGASA		GMASA		GASAVNS	
	Obj. Func.	**P. Eval**	**Obj. Func.**	**P. Eval**	**Obj. Func.**	**P. Eval**	**Obj. Func.**	**P. Eval**	**Obj. Func.**	**P. Eval**
1	3284	301865	3284	158362	3284	35691	3284	56731	3284	41849
2	3300	300806	3284	158808	3284	35702	3284	55969	3284	41434
3	3284	301336	3284	159014	3284	35801	3284	56300	3284	41368
4	3284	301957	3284	158402	3284	35691	3284	57072	3284	42308
5	3284	301573	3284	158732	3284	35844	3284	56857	3284	41017
6	3316	301703	3284	158764	3284	35687	3284	56597	3284	40918
7	3284	301860	3284	158147	3284	35911	3284	56344	3284	41385
8	3316	301195	3284	158366	3284	35827	3284	55855	3284	41765
9	3284	301853	3284	158572	3284	35889	3284	56402	3284	41734
10	3284	301291	3284	159672	3284	35879	3284	56538	3284	41075

Table 5-11 Analysis results of Leu et al. (2000) for the second project

	GA		GASA		HGASA		GMASA		GASAVNS	
	Obj. Func.	**P. Eval**	**Obj. Func.**	**P. Eval**	**Obj. Func.**	**P. Eval**	**Obj. Func.**	**P. Eval**	**Obj. Func.**	**P. Eval**
1	32200	301320	32200	158106	32200	34858	32200	51483	32200	38598
2	32200	302239	32200	159071	32200	34778	32200	51817	32200	38646
3	32200	300903	32200	159119	32200	34821	32200	51594	32200	38898
4	32200	301771	32200	158984	32200	34740	32200	51566	32200	39141
5	32200	301027	32200	158592	32200	34641	32200	51678	32200	38832
6	32448	301205	32200	158883	32200	34768	32200	51473	32200	38755
7	32200	302161	32200	158086	32200	34714	32200	51336	32200	38685
8	32200	301828	32200	158242	32200	34585	32200	51533	32200	38737
9	32200	301671	32200	159098	32200	34658	32200	51344	32200	38948
10	32200	301385	32200	159163	32200	34779	32200	52095	32200	39068

Table 5-12 Analysis results of the project of Lu and Lam (2008)

	GA		GASA		HGASA		GMASA		GASAVNS	
	Obj. Func.	**P. Eval**	**Obj. Func.**	**P. Eval**	**Obj. Func.**	**P. Eval**	**Obj. Func.**	**P. Eval**	**Obj. Func.**	**P. Eval**
1	5240	302123	5240	159280	5240	34634	5240	51683	5240	38816
2	5240	301403	5240	158878	5240	34540	5240	51285	5240	38401
3	5240	302061	5240	158478	5240	34547	5240	51226	5240	38947
4	5240	301195	5240	159282	5240	34700	5240	51357	5240	38662
5	5240	302158	5240	158257	5240	34753	5240	51298	5240	38702
6	5240	301806	5240	158038	5240	34555	5240	51598	5240	38714
7	5240	302019	5240	158201	5240	34774	5240	50809	5240	39407
8	5240	301319	5240	158263	5240	34601	5240	51167	5240	38708
9	5240	302133	5240	158940	5240	34911	5240	51486	5240	38552
10	5240	302259	5240	158569	5240	34573	5240	51526	5240	38894

Table 5-13 Analysis results of the project of Woodworth and Willie (1975)

	GA		GASA		HGASA		GMASA		GASAVNS	
	Obj. Func.	**P. Eval**	**Obj. Func.**	**P. Eval**	**Obj. Func.**	**P. Eval**	**Obj. Func.**	**P. Eval**	**Obj. Func.**	**P. Eval**
1	2512	301542	2512	158468	2512	36476	2512	57424	2512	41303
2	2512	300598	2512	158502	2512	36178	2512	56992	2512	41628
3	2512	301283	2512	158985	2512	36195	2512	56855	2512	41940
4	2512	302508	2512	159099	2512	36321	2512	57307	2512	40963
5	2512	301752	2512	158649	2512	36183	2512	57757	2512	41419
6	2512	300783	2512	158694	2512	36248	2512	57305	2512	41780
7	2512	301831	2512	158908	2512	36413	2512	58047	2512	41285
8	2512	301685	2512	158679	2512	36515	2512	57192	2512	41025
9	2512	302153	2512	159456	2512	36155	2512	57272	2512	41732
10	2512	301765	2512	158513	2512	36448	2512	56908	2512	41544

5.3 Conclusion

In this chapter resource leveling capability of the generated meta-heuristic algorithms are measured. Four of the algorithms successfully converged into known best solutions in all of the trials within a reasonable computation duration. In addition to this, one of the known best solutions is improved. Only the schedule of this solution is given in order to limit the number of schedule tables. The other schedules are exactly the same with the ones given in the literature.

The algorithms are capable of performing resource leveling for multiple resources and for any type of logical network relationships. This increases the usability of the algorithms for real construction projects.

The computation duration is measured as only 10 seconds for one trial of GASA in a 2.4 GHz CPU for 13 activity project. GASA is the slowest algorithm among the four algorithms which converge into known best solution in all trials. The reasonable computation time implies that the resource leveling of large construction projects can be performed by the meta-heuristic algorithms. The improvements in computer technology will even make the analysis duration shorter.

HGAQSA is the most successful resource leveling meta-heuristic algorithm among the analyzed meta-heuristic algorithms. GMASA, GASAVNS and GASA are also obtained global optimum. However, their convergence speed is not as fast as HGAQSA.

Resource leveling problem is an important problem type where there is not any method which guarantees obtaining the global optimum of the problem. As this is the case, the success of the meta-heuristic algorithms becomes more important.

CHAPTER 6

SINGLE MODE RESOURCE CONSTRAINT SCHEDULING

Single mode resource constraint scheduling problem (SRCPSP) deals with project scheduling problem in which activities have only one execution mode and the available resources are restricted. In this chapter SRCPSP type problems are analyzed by using three GA based and PSO meta-heuristic algorithms. The algorithms adapted for SRCPSP are GA, GASA, HGASA and PSO. As the test set 30, 60, 90 and 120-activity projects are used obtained from http://129.187.106.231/psplib/main.html.

Problem sets with 30, 60 and 90-activity projects consists of 480 projects in each data set. 120-activity data set contains 600 projects. There are four renewable resources in each project with a certain maximum available amount. None of the projects require non-renewable resource. The aim of the analysis is to complete the project in shortest duration without overriding the resource constraints and network relationships.

In the next subchapter the genetic algorithm based and PSO meta-heuristic algorithms and the solution algorithm of SRCPSP is briefly described. Analysis of 30, 60, 90 and 120-activity projects are given in the following subchapters.

6.1 Meta-heuristic Algorithms for the Solution of SRCPSP

SRCPSP is analyzed by genetic algorithm based methods, GA, GASA and HGASA, and PSO. For the analysis of SRCPSP by genetic algorithm based methods, gene represents the priority of the activity. This type of representation differs from the representations of crashing alternative and delay of activity. The priorities of the activities are represented by integer numbers starting from 1 to the activity number of the project. The smaller number means higher priority of that activity. If there is a resource limitation for the execution of more than one activities at the same time, execution of activities are determined according to the priorities of the activities.

Genetic algorithm for SRCPSP consists of generation of population, crossover, mutation and natural selection. In the following sections the operators of genetic algorithm for SRCPSP is explained.

Genes of the individuals represents the priority of the individual in SRCPSP type problems. For this reason, population should be generated accordingly. Gene of an individual consists of randomly sequenced integer numbers from 1 to the activity number of the project. Initially genes of the individuals are formed by sequenced integer numbers from 1 to the activity number of the project which is shown in Figure 6.1.

1	2	3	4	5	6	7	8

Figure 6.1 Initial sequencing of an individual of 8-Activity project.

The randomly sequencing of the activity priorities are achieved shifting the positions of priority numbers of activities in pairs. Couples are determined by generating two random integer numbers between 1 and project size. In Figure 6.2 shifted activity priorities of the individual is shown if the generated random numbers are 2 and 5.

1	5	3	4	2	6	7	8

Figure 6.2 Shifted priority of 8-Activity project.

Shifting of priority rules is performed five times the project-size. In this example, the shifting process is repeated 40 times for the 8-Activity project. During the shifting process CPM restrictions are not taken into account and as a result pure randomly generated activity priorities are obtained. Activity priorities are corrected by considering CPM restrictions. If an activity has higher priority compared with its predecessors, meaning an override of CPM logical relationships, the activity priorities of the two activities are exchanged. After the exchange, the check of the CPM logic starts from the first activity of the network. The search procedure is terminated if and only if there is not any override of CPM logical relationship in whole search of the network. This process is repeated for every individual of the population and population generation operation of SRCPSP is completed.

Mutation operator is also adapted for the gene representation of priority rules. Mutation operator exchanges the priorities of a couple which is formed randomly. In order to form the couple, two random integer numbers are generated and priorities of the two activities are exchanged. After the priority shifting in case of an override of CPM logic, activity priorities are checked and necessary corrections are performed. After the mutation operator, one schedule generation is counted although same activity priorities are obtained or the mutation is rejected.

Determination of genes to be mutated is exactly the same with the previous methods. For every individual, a randomly generated number is assigned. The individuals, whose randomly generated number is less than the mutation ratio are selected for the mutation.

Crossover operator is also adopted for the activity priority representation. Main problem of the crossover is the preservation of uniqueness of activity priorities. For this reason, crossover operator is significantly different than the one for the binary representation. In Figure 6.3, a couple mated for crossover for 8-activity project is shown. Crossover operation will be performed after the 5^{th} activity.

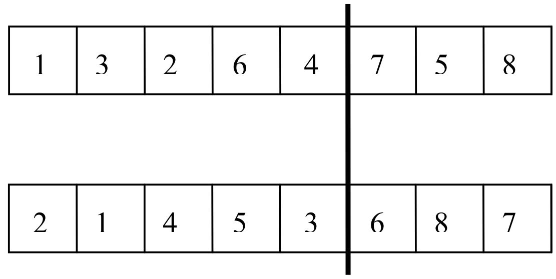

Figure 6.3 Mated couples for crossover operator

In classical crossover operator, the remaining genes are exchanged. However, in the activity priority representation such a gene exchange may cause prevention of representation of some of the activity priorities. In the crossover operation shown in Figure 6.3, if genes are exchanged conventionally for the child 1, both activity 4 and activity 6 would have the priority 6. In order to prevent such conflicts the exchanged genes are corrected.

Same priority assignments are detected in the two exchanged gene particles. In Figure 6.4 representation of detected same priorities are shown in the middle box sequence. The “0”(s) in the box represents different priority representations which will cause problems if they are directly exchanged. In the last box sequence, different priority assignments of the two genes to be exchanged are represented. The remaining places are filled with “0”(s). Different activity priorities would be exchanged in order to preserve the condition that any activity can not be assigned to same priority with another activity. In Figure 6.5 correction of the genes to be shifted is shown. Same genes are preserved and the different priority assignments in the genes are exchanged. This exchange procedure is shown in the first box sequence of Figure 6.5. The corrected gene sequences are cross-mated and attached to the parent’s genes and two new gen combinations are formed. The process is illustrated in Figure 6.5. The new gene combinations are checked for any possible overridden CPM logical rules. If there is any conflict, they are corrected.

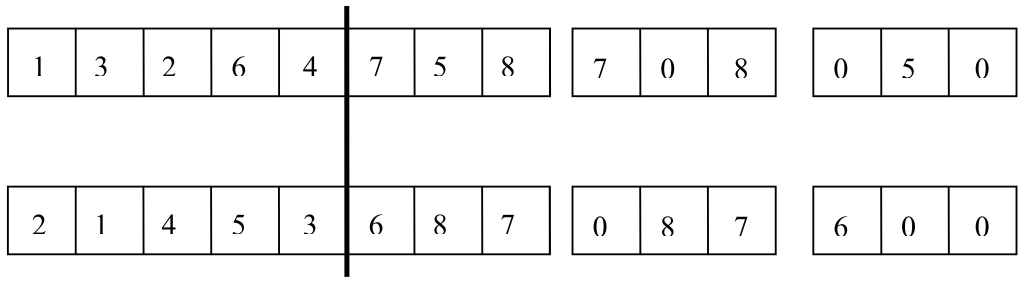

Figure 6.4 Crossover operator for the activity priority representation

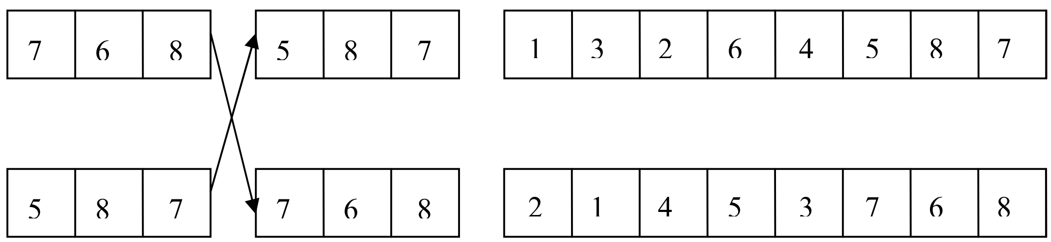

Figure 6.5 Exchange of genes within the crossover operator

Final operator is the natural selection operator. For the selection of genes to be terminated roulette-wheel selection method is applied. Same number of individual generated by the crossover operator is terminated by the natural selection operator. In order to protect the best gene, elitist natural selection operator is applied and the best gene is always preserved. Genes giving shorter project duration are assigned more probability of survival than the genes giving longer project duration.

SRCPSP is also analyzed by PSO. In PSO meta-heuristic algorithm, activity priorities are represented by the position. Positions of each individual are initialized by the same method applied for GA based algorithms. The randomly generated positions are evaluated and best value of the population is obtained. Individual's current project durations will be their initial individual best. Velocities of the particles are computed and new positions are computed by adding the velocities.

The updated positions contain floating numbers as the velocity computation is based on floating numbers. The range of the positions are limited in the range between zero and project size. In order to determine activity priorities, the positions are listed in ascending order. The smallest position will have the highest priority and the highest position will have the lowest priority. After ordering the positions, activity priorities are obtained which are than corrected for CPM logical relationships. The individuals are evaluated and if there is any improvement, population best and individuals' best values are updated. The iteration is continued until stopping criteria is met.

Implementations of the algorithms are explained by means of pseudocodes. The steps of the algorithms are introduced detailed in text. Pseudo code of the GA algorithm is given below;

begin
Set the population size, Ps equal to no of activities of the Project Ns, the crossover probability, Pc is set as 0.4, the mutation probability, Pm is set as 0.1 and the stopping condition is Nc^2model generation.
Generate population randomly
Check for the Network logical restrictions
Model generation counter i is set as 0
While i < Nc^2:
Select individuals from P for mutation with probability Pm.
Perform mutation and accept any mutation.
Check for the Network logical restrictions
Select chromosomes from P for crossover with probability Pc.
Randomly match the individuals selected for crossover
Perform crossover
Check for the Network logical restrictions
Assign probability of survival with the inverse of project completion duration, 1/Pd
Randomly select the individuals by roulette wheel natural selection algorithm
Terminate equal number individuals produced in crossover
i = i + 1
end

The evaluation of the genes is performed by considering the CPM logical relationships, priority rules and resource constraints. By using the priority assignment of the gene, early start date of the activity is computed by considering only the predecessors of the activity. If the resource constraints are not violated start date is accepted and the finish date of the activity is computed. If there is a violation of resource limits, the activity is delayed 1 day and the resource usages are checked. The delay of the activity is continued until the resource violation is prevented. After computing the activity finish time, another activity with the next priority is taken into account.

The operators mentioned above form GA for SRCPSP. In order to improve the solution capability of GA, mutation operator is modified by SA and the resulted method is called GASA. Modified mutation operator accepts or rejects a mutation based on the mutation's results and the temperature. If the mutation is beneficial, it is certainly accepted. On the other hand, harmful mutations are accepted or rejected based on the temperature and the amount of elongation of project duration. In the earlier iterations the probability of acceptance of harmful mutations are higher. The pseudo-code of GASA is given as;

begin
Set the population size, Ps equal to no of activities of the Project Ns, the crossover probability, Pc is set as 0.3, the mutation probability, Pm is set as 0.4 and the stopping condition is Nc * Nc model generation, Boltzmann Constant is set to the project completion duration with unlimited resources.
Set temperature T initially to Nc^2
Generate population randomly
Check for the Network logical restrictions
Model generation counter i is set as 0
While $i < Nc^2$:
Select individuals from P for mutation with probability Pm.
Perform mutation
Check for the acceptance criteria and accept if $Rn < \exp((f_i - f_i')*BC/T)$ where Rn is a randomly generated number between 0 and 1, f_i is the project duration before mutation, fi' is the project duration after the mutation.
Check for the Network logical restrictions
Select chromosomes from P for crossover with probability Pc.
Randomly match the individuals selected for crossover
Perform crossover
Check for the Network logical restrictions
Assign probability of survival with the inverse of project completion duration, 1/Pd
Randomly select the individuals by roulette wheel for natural selection
Terminate equal number of individuals produced in crossover
i = i + 1
T = T -1
end

Both GA and GASA do not have an advanced local search capability. In order to improve GASA, a local search operator is embedded into GASA and hybrid meta-heuristic algorithm based on GA and SA (HGASA) is formed. With embedded local search operator the algorithm seeks better neighbors of the best gene and randomly selected genes. Local search operator is executed after several generation of GA. In order to limit the computational burden, the whole population is not exposed to local search. Best gene is exposed to local search in order to obtain better gene representations. Local search of only best gene may cause being stuck into local minima if best gene is not close to global optima. For this reason, some of the randomly selected genes are exposed to local search. The local search gradually improves the population's overall gene quality and increases the probability of producing better genes at the end of crossover operator. The pseudo-code of HGASA is given as;

begin
Set the population size, Ps equal to no of activities of the Project Ns, the crossover probability, Pc is set as 0.3, the mutation probability, Pm is set as 0.4 and the stopping condition is Nc * Nc ^(1/2) model generation, Boltzmann Constant is set to the project completion duration with unlimited resources.
Set temperature T initially to Nc^2, Local search period, Ls is set as 5.
Local search amount, La is set as 4*Nc, and searched individual number Si, is set as 4.
Generate population randomly
Check for the Network logical restrictions
Model generation counter i is set as 0
While i < $Nc^{3/2}$:
Select individuals from P for mutation with probability Pm.
Perform mutation
Check for the acceptance criteria and accept if Rn < $\exp((f_i - f_i')*BC/T)$ where Rn is a randomly generated number between 0 and 1, f_i is the project duration before mutation, fi' is the project duration after the mutation
Check for the Network logical restrictions
Select chromosomes from P for crossover with probability Pc.
Randomly match the individuals selected for crossover
Perform crossover
Check for the Network logical restrictions
Assign probability of survival with the inverse of project completion duration, 1/Pd
Randomly select the individuals by roulette wheel for natural selection
Terminate equal number of individuals produced in crossover
Check if hybrid Local search condition satisfied (i % Ls = 0)
    if true
    j = 0
    Select current best individual and randomly Si - 1 individuals
    While j < La:
    Perform mutation
    Check for the acceptance criteria and accept mutation if Rn < $\exp((f_i - f_i')*CB*BC/T)$ where CB is the project completion duration of the current best individual
    Check for the Network logical restrictions
    j = j + 1
    end
i = i + 1
T = T -1
end

Local search continuously mutates the selected gene. The decision of the acceptance of the mutations is taken based on SA. The decision for the acceptance of the mutations has vital importance since too easily acceptance of harmful mutations may cause termination of good genes and too strict acceptance criteria may prevent escaping from local optima.

6.2 Test Problems

For the test problems, some of the well known resource constraint schedule problems found on the literature is used. The simplest one is the 8-Activity project of a two-span bridge construction project (Toklu 2002). The resource constraint project has 72 possible scheduling alternatives which can be considered to be a very small search space. If the resources are unlimited the shortest project completion date is 75 days. With the limited resources MS Project gives the shortest project duration 114 days and Primavera P3 gives 139 days. By changing the order of activities the Toklu had achieved 131 days of project completion with Primavera P3. The possible earliest project completion duration of the project with unlimited resources is 108 days.

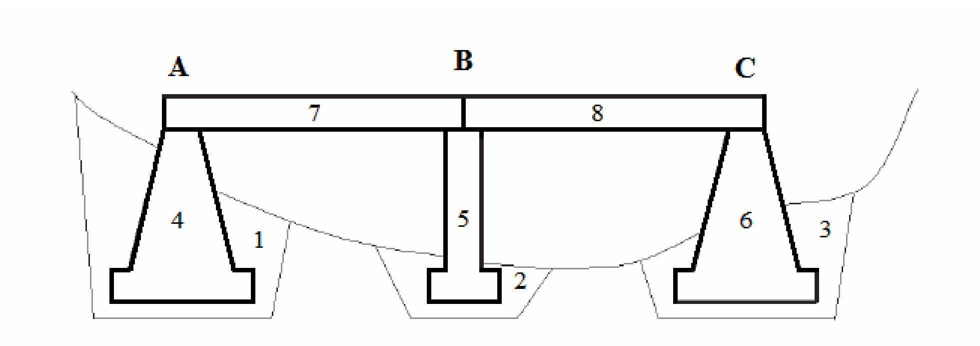

Figure 6.6 Two-span bridge with the construction activities (Toklu 2002)

Two-span bridge and the network diagram also representing resource demands of the activities are given in Figure 6.6 and Figure 6.7 respectively.

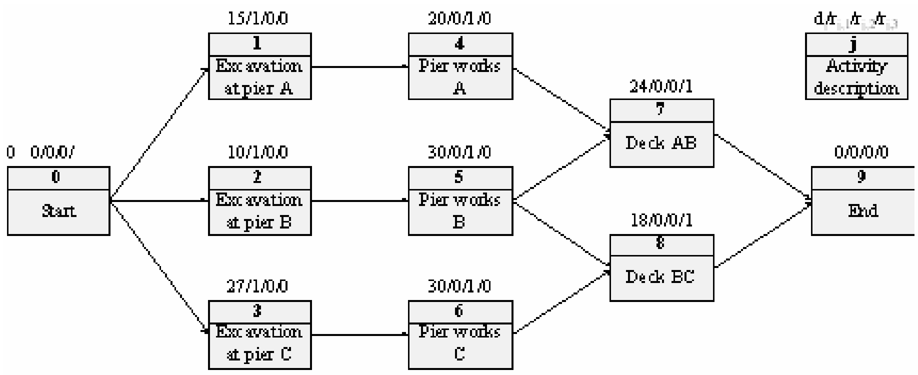

Figure 6.7 Network diagram of the example project (Toklu 2002)

The project is analyzed by GA, GASA and HGASA and the following results are obtained.

Table 6-1 Analysis of the 8-Activity project by GASA

Analysis/Schedule	**10**	**50**	**250**	**500**
1	113	108	108	108
2	108	108	108	108
3	114	114	108	108
4	108	108	108	108
5	108	108	108	108
6	123	113	108	108
7	113	113	108	108
8	114	108	108	108
9	108	108	108	108
10	114	114	108	108

At the end of the 10 run, following priority assignments which give the 108 days project completion duration are obtained. Although the priority assignments are different, exactly the same schedule is obtained. The reason of this is the scheduling algorithm which tries to execute the activities as soon as possible. Activity B2 is labeled as 4 in the priority sequence. In the first solution B2 has the 4th priority among the activities and in the second solution B2 has the second priority. However, activities labeled by 1 and 5 which are A1 and C1 do not demand the same resources with activity B2 and they do not cause any delay for the activity B2.

Table 6-2 Activity priority lists of the best solutions

3	1	5	4	2	7	6	8
3	4	1	5	2	6	7	8
3	4	1	2	7	5	6	8
3	1	4	2	5	6	8	7
3	1	5	4	2	7	6	8
3	1	4	2	5	6	7	8
3	4	1	5	2	6	8	7
3	1	4	5	2	6	8	7
3	1	4	2	5	6	8	7
3	1	4	2	5	7	6	8

The 8-Activity project is good enough to represent the solution capability of the GA, GASA, HGASA and PSO when the analyses results are compared by the results of most commonly used commercial software. Primavera converged into 113 days schedule while MS Project converged into 119 days of project duration. In order to have more idea about the capability of the software the test problem is enlarged and a 5-span bridge construction with limited resources is analyzed. The project consists of 42 activities. Logical relationships between the activities, activity durations and resource usages of the activities are given in Table 6-3 (Bartusch, 1983).

Available resources are only 1 for each resource type during the construction period. In addition to the standard precedence constraints given in Table 6-3, additional constraints are also defined.

They are listed as;

- The formworks must start at least six days after the beginning of the erection of the temporary housing.
- Removal of the temporary housing can start at most two days before the end of the last masonry work
- Delivery of the performed bearers occurs exactly 30 days after the beginning of the project

Table 6-3 Activity properties of the 42-Activity project

														Resources			
Act ID	**No Suc**	**Successors**							**Dur**	**R1**	**R2**	**R3**	**R4**	**R5**	**R6**	**R7**
1	7	2	3	4	5	6	7	10	0	0	0	0	0	0	0	0
2	1	11							4	1	0	0	0	0	0	0
3	1	12							2	1	0	0	0	0	0	0
4	1	8							2	1	0	0	0	0	0	0
5	1	9							2	1	0	0	0	0	0	0
6	1	15							2	1	0	0	0	0	0	0
7	1	16							5	1	0	0	0	0	0	0
8	1	13							20	0	1	0	0	0	0	0
9	1	14							13	0	1	0	0	0	0	0
10	1	41							10	0	0	0	0	0	0	0
11	1	17							8	0	0	1	0	0	0	0
12	1	18							4	0	0	1	0	0	0	0
13	1	19							4	0	0	1	0	0	0	0
14	1	20							4	0	0	1	0	0	0	0
15	1	21							4	0	0	1	0	0	0	0
16	1	22							10	0	0	1	0	0	0	0
17	1	23							1	0	0	0	1	0	0	0
18	1	24							1	0	0	0	1	0	0	0
19	1	25							1	0	0	0	1	0	0	0
20	1	26							1	0	0	0	1	0	0	0
21	1	27							1	0	0	0	1	0	0	0
22	1	28							1	0	0	0	1	0	0	0
23	1	29							1	0	0	0	1	0	0	0
24	1	30							1	0	0	0	0	0	0	0
25	1	31							1	0	0	0	0	0	0	0
26	1	32							1	0	0	0	0	0	0	0
27	1	33							1	0	0	0	0	0	0	0
28	1	34							1	0	0	0	0	0	0	0
29	1	36							16	0	0	0	0	1	0	0
30	2	36	37						8	0	0	0	0	1	0	0
31	2	37	38						8	0	0	0	0	1	0	0
32	2	38	39						8	0	0	0	0	1	0	0
33	2	39	40						8	0	0	0	0	1	0	0
34	1	40							20	0	0	0	0	1	0	0
35	5	36	37	38	39	40			2	0	0	0	0	0	1	0
36	1	42							12	0	0	0	0	0	1	0
37	1	44							12	0	0	0	0	0	1	0
38	1	44							12	0	0	0	0	0	1	0
39	1	44							12	0	0	0	0	0	1	0
40	1	43							12	0	0	0	0	0	1	0
41	1	44							10	0	0	0	0	0	0	0
42	1	44							15	0	0	0	0	0	0	1
43	1	44							10	0	0	0	0	0	0	1
44	0								0	0	0	0	0	0	0	0

The project is scheduled by considering the precedence constraints and the additional constraints. The analyses are repeated 10 times for each of the meta-heuristic algorithm and the results are tabulated in Table 6-4 to Table 6-7.

Table 6-4 Solution obtained by GA

Analysis No	**1000**	**5000**	**25000**
1	104	104	104
2	105	104	104
3	108	105	104
4	104	104	104
5	104	104	104
6	104	104	104
7	104	104	104
8	104	104	104
9	104	104	104
10	104	104	104

Table 6-5 Solution obtained by GASA

Analysis No	**1000**	**5000**	**25000**
1	104	104	104
2	104	104	104
3	104	104	104
4	104	104	104
5	104	104	104
6	104	104	104
7	104	104	104
8	104	104	104
9	104	104	104
10	104	104	104

Table 6-6 Solution obtained by HGASA

Analysis No	**1000**	**5000**	**25000**
1	104	104	104
2	107	104	104
3	104	104	104
4	104	104	104
5	104	104	104
6	105	104	104
7	104	104	104
8	104	104	104
9	106	104	104
10	105	104	104

Table 6-7 Solution obtained by PSO

Analysis No	**1000**	**5000**	**25000**
1	105	104	104
2	106	104	104
3	104	104	104
4	105	104	104
5	104	104	104
6	105	104	104
7	104	104	104
8	105	104	104
9	105	104	104
10	105	104	104

It can be seen from Table 6-4 to Table 6-6 that GA, GASA and HGASA converges to the same solution which is 104 days. The convergence of GASA and HGASA is faster than GA that at the end of the 5000th iteration GASA and HGASA had obtained 104 days of schedule while GA obtained same solution in 7 out of 10 trials. However, convergence of GASA is significantly faster than HGASA that GASA obtained 104-days solution at all trials after the 1000th run. PSO is also obtained 104 days schedule at the end of the analysis. However, convergence of PSO is not as fast as GASA. PSO converged into global optimum more or less the same rate with HGASA. The 42-activity project is not difficult enough to measure the meta-heuristic algorithms' convergence capability to the global optima. Schedule of 104-day solution is given in Table 6-8. GA, GASA, HGASA and PSO obtained the same construction schedule.

Table 6-8 Schedule of 42-activity project

Activity ID	Start	Finish	Duration	R1	R2	R3	R4	R5	R6	R7
1	6	10	4	1	0	0	0	0	0	0
2	0	2	2	1	0	0	0	0	0	0
3	4	6	2	1	0	0	0	0	0	0
4	2	4	2	1	0	0	0	0	0	0
5	10	12	2	1	0	0	0	0	0	0
6	12	17	5	1	0	0	0”	0	0	0
7	6	26	20	0	1	0	0	0	0	0
8	26	39	13	0	1	0	0	0	0	0
9	0	10	10	0	0	0	0	0	0	0
10	10	18	8	0	0	1	0	0	0	0
11	6	10	4	0	0	1	0	0	0	0
12	26	30	4	0	0	1	0	0	0	0
13	39	43	4	0	0	1	0	0	0	0
14	18	22	4	0	0	1	0	0	0	0
15	43	53	10	0	0	1	0	0	0	0
16	18	19	1	0	0	0	1	0	0	0
17	10	11	1	0	0	0	1	0	0	0
18	30	31	1	0	0	0	1	0	0	0
19	43	44	1	0	0	0	1	0	0	0
20	22	23	1	0	0	0	1	0	0	0
21	53	54	1	0	0	0	1	0	0	0
22	19	20	1	0	0	0	1	0	0	0
23	11	12	1	0	0	0	0	0	0	0
24	31	32	1	0	0	0	0	0	0	0
25	44	45	1	0	0	0	0	0	0	0
26	23	24	1	0	0	0	0	0	0	0
27	54	55	1	0	0	0	0	0	0	0
28	20	36	16	0	0	0	0	1	0	0
29	12	20	8	0	0	0	0	1	0	0
30	44	52	8	0	0	0	0	1	0	0
31	52	60	8	0	0	0	0	1	0	0
32	36	44	8	0	0	0	0	1	0	0
33	60	80	20	0	0	0	0	1	0	0
34	30	32	2	0	0	0	0	0	1	0
35	36	48	12	0	0	0	0	0	1	0
36	52	64	12	0	0	0	0	0	1	0
37	64	76	12	0	0	0	0	0	1	0
38	92	104	12	0	0	0	0	0	1	0
39	80	92	12	0	0	0	0	0	1	0
40	78	88	10	0	0	0	0	0	0	0
41	48	63	15	0	0	0	0	0	0	1
42	92	102	10	0	0	0	0	0	0	1

The performance of GA, GASA and HGASA is tested by using the randomly generated resource constraint scheduling problems obtained from the http://129.187.106.231/psplib/main.html. Problem set consists of 30, 60, 90 and 120-Activity projects which have four limited nonrenewable resources. There are 600 120-Activity projects and 480 projects for each of the remaining project sizes. The projects are solved by GA, GASA, HGASA and PSO meta-heuristic algorithms.

Test problems obtained from PSPLIB are randomly generated projects on the basis of certain rules. The network diagram is based on successor relationships that an activity can have at least one successor and at most three successors. Duration of activities are also assigned randomly between 1 and 10 days. Resource demand of the activities may be limited by one resource type or an activity may require four resource types for its execution.

The problems are previously analyzed by many researchers which is also mentioned in the literature review part. Upper and lower bounds for the test problems are determined and if there is any improvement in these bounds, than the bounds of the test problem is updated.

In the literature, there are many algorithms for the determination of the lower bounds. Brucker and Knust (2000) mentions that *constructive* and *destructive* lower bounds can be derived for SRCPSP. Constructive bound is the length of the longest path in the network which is provided by solving relaxations of the SRCPSP. The relaxed problem is obtained by dropping resource constraints (Mingozzi et al. 1998).

Destructive bounds restrict a problem by setting a maximal objective function value and try to destruct the feasibility of this reduced problem. If there is not a schedule with make-span less than or equal to the maximal objective function, than the maximal objective function will be increased by one day (Brucker and Knust 2003).

Mingozzi et al. (1998) present a binary linear programming formulation of the project makespan minimization problem (Möhring et al. 1999). The relaxation methods for the lower bound computation can be listed as critical path bound, capacity bound, critical sequence bound, node packing bound, parallel-machine bound, extended node packing bound, generalized node packing bound, one-machine bound, two-machine bound, precedence bound 1- 2 and Lagrangian relaxation (Klein and Scholl 1999).

Upper bounds of the test problem are the shortest project make-spans which are obtained by no relaxation procedure.

6.3 Analysis of Test Problems

Success of the algorithms is evaluated by the closeness of the best values at the 1000th, 5000th and 50000th schedule. The aim of the limitation of schedule number is to obtain an acceptable solution within reasonable computation duration. Besides the 50000 evaluation, for the 120-Activity projects are iterated up to 1M evaluation in order to measure the capability of finding global optimum of the algorithms.

The analysis results deviate since the iterations are stopped before converging into global optimum. In order to measure the amount of deviation of the results, each analysis is repeated 10 times. Minimum and maximum project completion durations are recorded and mean of the results are computed. The analysis results are tabulated according to the problem type and solution algorithm. In addition to this, lower and upper bound comparisons are illustrated in separate tables.

Columns with headings formed by numbers represent the deviation of the current best solution achieved at that number of schedule generation from the best optimal solution obtained from the literature. Stopping error column represents the deviation of the overall best value achieved by the algorithm from the best optimal solution obtained from the literature. *No of optimum solution* column represents the number of solutions in the problem set which is equal to the best known solution.

The analysis results are grouped according to the problem type in which the comparison of the meta-heuristic algorithms will be easier.

The J30 problem set which consists of 480 projects with 30-activities, is the easiest problem set. As a result the error values of the whole algorithms are small and close to each other. First 1000 schedules of the algorithms are more or less the same. This shows that the algorithms do not have significant difference in initial convergence ability to the global optimum. However, the highest mean deviation is %1.432 which can be considered as reasonable.

Table 6-9 GA analysis results of J30 problem set

Analysis No	**1000**	**5000**	**25000**	**50000**	**Stopping Error**	**No of Opt Sol**
1	1,200	0,635	0,379	0,320	0,206	434
2	1,183	0,589	0,340	0,270	0,196	435
3	1,179	0,617	0,364	0,313	0,210	433
4	1,207	0,698	0,355	0,272	0,211	434
5	1,155	0,691	0,398	0,336	0,249	433
6	1,214	0,659	0,382	0,294	0,206	438
7	1,170	0,616	0,349	0,256	0,177	438
8	1,146	0,687	0,387	0,322	0,247	428
9	1,179	0,699	0,354	0,279	0,201	433
10	1,130	0,608	0,359	0,299	0,236	432
Mean	**1,176**	**0,650**	**0,367**	**0,296**	**0,214**	**433,8**

Table 6-10 GASA analysis results of J30 problem set

Analysis No	**1000**	**5000**	**25000**	**50000**	**Stopping Error**	**No of Opt Sol**
1	1,059	0,510	0,240	0,164	0,140	445
2	1,091	0,526	0,202	0,168	0,135	449
3	1,124	0,555	0,241	0,171	0,124	451
4	1,068	0,505	0,236	0,165	0,126	447
5	1,081	0,525	0,216	0,172	0,147	447
6	1,079	0,531	0,234	0,148	0,118	451
7	1,079	0,492	0,223	0,150	0,108	454
8	1,043	0,550	0,248	0,161	0,138	446
9	1,141	0,573	0,220	0,153	0,107	452
10	1,011	0,479	0,221	0,175	0,148	446
Mean	**1,078**	**0,525**	**0,228**	**0,162**	**0,129**	**448,8**

Table 6-11 HGASA analysis results of J30 problem set

Analysis No	**1000**	**5000**	**25000**	**50000**	**Stopping Error**	**No of Opt Sol**
1	1,125	0,380	0,094	0,072	0,049	465
2	1,135	0,345	0,100	0,060	0,051	464
3	1,121	0,351	0,090	0,064	0,044	467
4	1,150	0,379	0,088	0,055	0,047	465
5	1,180	0,373	0,100	0,070	0,070	463
6	1,201	0,369	0,088	0,063	0,042	468
7	1,105	0,393	0,098	0,060	0,044	467
8	1,188	0,358	0,106	0,057	0,044	466
9	1,116	0,390	0,100	0,081	0,062	462
10	1,145	0,379	0,100	0,055	0,044	467
Mean	**1,147**	**0,372**	**0,096**	**0,064**	**0,050**	**465,4**

Table 6-12 PSO analysis results of J30 problem set

Analysis No	**1000**	**5000**	**25000**	**50000**	**Stopping Error**	**No of Opt Sol**
1	1,377	0,971	0,670	0,555	0,545	391
2	1,463	1,018	0,714	0,597	0,571	387
3	1,468	1,016	0,697	0,570	0,567	387
4	1,435	1,012	0,706	0,620	0,611	378
5	1,479	0,999	0,704	0,594	0,576	383
6	1,435	0,994	0,703	0,615	0,610	379
7	1,381	0,999	0,689	0,565	0,562	383
8	1,430	1,019	0,671	0,557	0,537	388
9	1,428	1,034	0,690	0,571	0,561	384
10	1,421	1,001	0,700	0,578	0,557	391
Mean	**1,432**	**1,007**	**0,694**	**0,582**	**0,570**	**385,1**

Performance of the algorithms deviates when the schedule number increases. GA has less significance improvement when compared with the GASA and HGASA. However, GA is better than PSO. The GA algorithm is stopped at the end of 130000 schedules and ends up with 433,8 average optimum solution out of 480 projects. However, GASA obtains 448,8 average optimum solutions in 85000 schedules and HGASA obtains 465,4 average optimum solutions in 85000 schedules. In addition to this, average errors of the algorithms shows a similar trend when the stopping criteria is met. The mean deviations of the GA, GASA and HGASA are 0,214, 0,129 and 0,050 respectively. Even GA can be considered as successful. On the other hand, PSO could not improve the initial convergence and at the end of 100.000 schedule evaluation 0,570 mean error is obtained. This means that PSO get stuck into local minima and could obtained only 0.012% of improvement in 50.000 evaluation. As a result of this PSO can obtain only 385,1 optimum solution on the average. The detailed analysis results are given in Table 6-9 to Table 6-12. In Table 6-13, average deviation of the meta-heuristic methods found in the literature from the optimal makespan is represented. In Table 6-13 the deviation results are given from the first analysis out of ten.

When the GA, PSO and GASA are compared with the results in Table 6-13, the three algorithms do not have a noticeable success. However, HGASA obtains %0,07 average error and takes the 13th position after Tormos and Lova (2001).

Table 6-13 Average deviation of the J30 from the optimal makespan

Algorithm	**Reference**	**Max. # schedules**		
		1000	5000	50000
Hybrid scatter	Ranjbar et al. (2009)	0.10	0.03	0.00
GA, TS, path relinking	Kochetov and Stolyar (2003)	0.10	0.04	0.00
GAPS	Mendes et al. (2009)	0.06	0.02	0.01
Scatter Search—FBI	Debels et al. (2006)	0.27	0.11	0.01
GA—DBH	Debels and Vanhoucke (2005)	0.15	0.04	0.02
Hybrid GA	Valls et al. (2008)	0.27	0.06	0.02
GA—FBI	Valls et al. (2005)	0.34	0.20	0.02
GA-forw.-back. —FBI	Alcaraz et al. (2004)	0.25	0.06	0.03
GA	Alcaraz and Maroto (2001)	0.33	0.12	–
Sampling + BF/FB	Tormos and Lova (2003)	0.25	0.13	0.05
Tabu Search	Nonobe and Ibaraki (2002)	0.46	0.16	0.05
Sampling + BF	Tormos and Lova (2001)	0.30	0.16	0.07
HGASA—priority list	This study	1.13	0.38	0.07
GA—self-adapting	Hartmann (2002)	0.38	0.22	0.08
GA—activity list	Hartmann (1998)	0.54	0.25	0.08
Sampling + BF	Tormos and Lova (2003b)	0.30	0.17	0.09
TS—activity list	Klein (2000)	0.42	0.17	
Sampling—FBI	Valls et al. (2005)	0.46	0.28	0.11
SA—activity list	Bouleimen and Lecocq (2003)	0.38	0.23	–
GA—late join	Coelho and Tavares (2003)	0.74	0.33	0.16
Sampling—adaptative	Schirmer (2000)	0.65	0.44	–
TS—schedule scheme	Baar et al. (1998)	0.86	0.44	–
GASA—priority list	This study	1.06	0.51	0.16
Sampling—adaptative	Kolisch and Drexl (1996)	0.74	0.53	–
GA—random key	Hartmann (1998)	1.03	0.56	0.23
Sampling—LFT	Kolisch (1996)	0.83	0.53	0.27
Sampling—global	Coelho and Tavares (2003)	0.81	0.54	0.28
GA—priority list	This study	1.21	0.64	0.32
PSO—priority list	This study	1.38	0.97	0.56

60-Activity project set is more difficult than the 30-Activity set; as a result the deviation amounts obtained in this analysis are larger. However, the convergence of the algorithms is very similar to the previous analysis. The deviations of project duration from the lower bounds in percentages are given in Table 6-14 to Table 6-17. At the end of the 1000th schedule there is no significant difference between the performances of the algorithms where all meta-heuristic algorithms have approximately 5.5% mean deviation. At the end of the 5000th schedule GASA has an apparent improvement when compared with the 1000th schedule. However, at the end of the 250000th schedule HGASA catches the GASA and gives slightly better results. On the other hand, there is not significant difference between the GASA and HGASA. GA and PSO could not improve their initial results significantly as a result of this; worst results are obtained with these two algorithms. GA can improve initial results slowly which is costly if near-optimum solutions are required. PSO can not improve its results and it is better to stop iteration after a certain point. When the results of 25000 and 50000 are examined, it is seen that there is very little improvement. This means that the algorithm get stuck into local minima. The analysis is stopped after the 150000 schedules.

Table 6-14 GA analysis results of J60 problem set with lower bound

Analysis No	**1000**	**5000**	**25000**	**50000**	**250000**	**Stopping Error**	**No of Opt Sol**
1	5,479	4,714	4,190	3,989	3,713	3,677	344
2	5,515	4,594	4,100	3,985	3,716	3,694	345
3	5,464	4,705	4,127	4,005	3,715	3,675	347
4	5,522	4,659	4,155	3,973	3,709	3,667	343
5	5,437	4,674	4,210	4,034	3,733	3,688	344
6	5,526	4,711	4,184	3,986	3,676	3,641	347
7	5,456	4,727	4,217	4,033	3,662	3,627	347
8	5,532	4,693	4,156	4,003	3,743	3,716	345
9	5,446	4,654	4,190	4,016	3,728	3,702	342
10	5,496	4,717	4,229	4,053	3,755	3,716	343
Mean	**5,487**	**4,685**	**4,176**	**4,008**	**3,715**	**3,680**	**344,7**

Table 6-15 GASA analysis results of J60 problem set with lower bound

Analysis No	1000	5000	25000	50000	250000	Stopping Error	No of Opt Sol
1	5,467	4,098	2,925	2,713	2,459	2,448	353
2	5,460	4,170	2,946	2,746	2,442	2,425	354
3	5,456	4,168	2,989	2,788	2,466	2,450	354
4	5,508	4,132	3,000	2,730	2,441	2,420	355
5	5,565	4,104	2,976	2,772	2,474	2,448	355
6	5,566	4,195	2,941	2,778	2,487	2,455	355
7	5,577	4,203	2,905	2,682	2,463	2,449	353
8	5,474	4,145	2,953	2,741	2,411	2,385	355
9	5,447	4,132	2,942	2,734	2,452	2,418	353
10	5,455	4,161	2,962	2,767	2,499	2,475	354
Mean	**5,497**	**4,151**	**2,954**	**2,745**	**2,459**	**2,437**	**354,1**

Table 6-16 HGASA analysis results of J60 problem set with lower bound

Analysis No	1000	5000	25000	50000	250000	Stopping Error	No of Opt Sol
1	5,535	4,466	3,103	2,686	2,388	2,361	355
2	5,480	4,465	3,129	2,705	2,387	2,334	355
3	5,490	4,522	3,088	2,720	2,398	2,364	354
4	5,493	4,480	3,092	2,644	2,364	2,339	354
5	5,516	4,472	3,159	2,770	2,392	2,365	354
6	5,420	4,499	3,140	2,745	2,460	2,409	355
7	5,467	4,471	3,125	2,732	2,405	2,370	353
8	5,528	4,480	3,110	2,763	2,384	2,366	355
9	5,559	4,445	3,103	2,712	2,368	2,333	354
10	5,489	4,521	3,143	2,696	2,406	2,363	354
Mean	**5,498**	**4,482**	**3,119**	**2,717**	**2,395**	**2,360**	**354,3**

Table 6-17 PSO analysis results of J60 problem set with lower bound

Analysis No	**1000**	**5000**	**25000**	**50000**	**Stopping Error**	**No of Opt Sol**
1	5,452	5,037	4,634	4,449	4,173	323
2	5,455	5,027	4,685	4,507	4,220	321
3	5,472	5,082	4,655	4,521	4,225	325
4	5,498	5,042	4,665	4,506	4,225	323
5	5,435	5,014	4,611	4,469	4,187	325
6	5,510	5,078	4,682	4,529	4,224	324
7	5,488	5,066	4,629	4,458	4,189	323
8	5,401	5,001	4,624	4,500	4,207	324
9	5,533	5,108	4,668	4,485	4,225	322
10	5,548	5,079	4,662	4,486	4,242	323
Mean	**5,479**	**5,053**	**4,651**	**4,491**	**4,212**	**323,3**

In Table 6-18, analysis results of different researches of the same problem set is represented for the first trial of the meta-heuristic algorithms. It is seen that HGASA, GASA, GA and PSO all have better results than the previous studies. GA based meta-heuristic algorithms are stopped at the end of the 325000th schedule, while PSO is stopped at the end of 150000th schedule.

Although it's slow convergence PSO represent better results than the previous studies obtained from the literature. The algorithm stopped earlier than the GA based algorithm because after a certain number of iterations, the algorithms convergence significantly slows down. Number of global optima found by PSO is significantly less than the other algorithms.

Table 6-18 Average deviation of the J60 from the optimal makespan

Algorithm	**Reference**	**Max. # schedules**		
		1000	5000	50000
Hybrid scatter	Ranjbar et al. (2009)	11.59	11.07	10.64
GAPS	Mendes et al. (2009)	11.72	11.04	10.67
GA-DBH	Debels and Vanhoucke (2005)	11.45	10.95	10.68
Scatter search-FBI	Debels et al. (2006)	11.73	11.10	10.71
GA-hybrid	Valls et al. (2008)	11.56	11.10	10.73
GA, TS-path relinking	Kochetov and Stolyar (2003)	11.71	11.17	10.74
GA-FBI	Valls et al. (2005)	12.21	11.27	10.74
GA-forw.-back. -FBI	Alcaraz et al. (2004)	11.89	11.19	10.84
GA-self-adapting	Hartmann (2002)	12.21	11.70	11.21
GA-activity list	Hartmann (1998)	12.68	11.89	11.23
Sampling-LFT, FBI	Tormos and Lova (2003b)	11.88	11.62	11.36
SA-activity list	Bouleimen and Lecocq (2003)	12.75	11.90	-
HGASA-priority list	This study	15.04	13.77	11.49
GASA-priority list	This study	14.97	13.31	11.52
TS-activity list	Nonobe and Ibaraki (2002)	12.97	12.18	11.58
Sampling-adaptative	Schirmer and Riesenberg (2000)	12.94	12.58	-
Sampling-adaptative	Kolisch and Drexl (1996)	13.51	13.06	-
GA-random key	Hartmann (1998)	14.68	13.32	12.25
GA-priority rule	Hartmann (1998)	13.30	12.74	12.26
Sampling-LFT	Kolisch (1996a)	13.59	13.23	12.85
Sampling-WCS	Kolisch (1996a, 1996b)	13.66	13.21	-
TS-schedule scheme	Baar et al. (1998)	13.80	13.48	-
GA-problem space	Leon and Ramamoorthy	14.33	13.49	-
Sampling-LFT	Kolisch (1996a)	13.96	13.53	12.97
GA-priority list	This study	14.98	14.09	13.20
Sampling-random	Kolisch (1995)	14.89	14.30	13.66
PSO-priority list	This study	14.98	14.55	13.99
Sampling-random	Kolisch (1995)	15.94	15.17	14.22

Table 6-19 GA analysis results of J60 problem set compared with UB

Analysis No	1000	5000	25000	50000	250000	Stopping Error	No of Opt Sol
1	3,379	2,645	2,149	1,959	1,700	1,666	348
2	3,414	2,533	2,067	1,958	1,705	1,684	348
3	3,367	2,638	2,090	1,974	1,700	1,663	351
4	3,420	2,594	2,117	1,944	1,696	1,658	347
5	3,339	2,606	2,168	2,001	1,718	1,677	346
6	3,426	2,644	2,142	1,957	1,666	1,633	351
7	3,357	2,659	2,173	2,000	1,651	1,619	350
8	3,431	2,629	2,117	1,972	1,728	1,704	348
9	3,349	2,589	2,151	1,984	1,715	1,691	346
10	3,394	2,648	2,185	2,020	1,741	1,703	346
Mean	**3,388**	**2,618**	**2,136**	**1,977**	**1,702**	**1,670**	**348,1**

Table 6-20 GASA analysis results of J60 problem set compared with UB

Analysis No	1000	5000	25000	50000	250000	Stopping Error	No of Opt Sol
1	3,366	2,067	0,970	0,772	0,536	0,526	377
2	3,361	2,134	0,988	0,802	0,520	0,505	383
3	3,360	2,132	1,029	0,841	0,540	0,525	378
4	3,405	2,099	1,040	0,790	0,519	0,499	380
5	3,464	2,072	1,018	0,828	0,550	0,526	379
6	3,462	2,157	0,986	0,834	0,561	0,531	380
7	3,472	2,165	0,952	0,743	0,540	0,526	378
8	3,376	2,111	0,997	0,798	0,492	0,467	382
9	3,349	2,100	0,988	0,793	0,529	0,498	377
10	3,358	2,127	1,006	0,824	0,574	0,552	379
Mean	**3,397**	**2,117**	**0,997**	**0,803**	**0,536**	**0,515**	**379,3**

Table 6-21 HGASA analysis results of J60 problem set compared with UB

Analysis No	1000	5000	25000	50000	250000	Stopping Error	No of Opt Sol
1	3,435	2,414	1,135	0,748	0,469	0,444	381
2	3,380	2,412	1,160	0,764	0,469	0,420	387
3	3,388	2,469	1,122	0,781	0,479	0,447	377
4	3,389	2,428	1,125	0,708	0,447	0,424	381
5	3,418	2,421	1,187	0,825	0,474	0,449	381
6	3,321	2,445	1,168	0,802	0,537	0,489	380
7	3,368	2,419	1,156	0,790	0,486	0,454	377
8	3,427	2,426	1,142	0,817	0,466	0,449	382
9	3,454	2,394	1,134	0,772	0,452	0,419	383
10	3,391	2,466	1,172	0,756	0,487	0,447	383
Mean	**3,397**	**2,429**	**1,150**	**0,776**	**0,477**	**0,444**	**381,2**

In Table 6-19 to Table 6-22, analysis results of J60 compared with the current best upper bound values obtained from the literature is shown. As expected the error ranges are smaller than the results of the lower bound comparison. Number of optimum solution is also increased as the makespan of upper bound solution set is usually higher. The algorithms could not improve any of the upper bound solutions.

Table 6-22 PSO analysis results of J60 problem set compared with UB

Analysis No	**1000**	**5000**	**25000**	**50000**	**Stopping Error**	**No of Opt Sol**
1	3,354	2,959	2,574	2,400	2,140	324
2	3,355	2,949	2,625	2,455	2,184	323
3	3,371	3,001	2,596	2,467	2,189	326
4	3,397	2,963	2,606	2,454	2,189	325
5	3,337	2,936	2,553	2,420	2,153	327
6	3,410	2,999	2,623	2,479	2,188	326
7	3,384	2,986	2,570	2,409	2,154	325
8	3,305	2,924	2,566	2,449	2,171	326
9	3,429	3,026	2,608	2,434	2,187	324
10	3,442	2,998	2,603	2,436	2,204	325
Mean	**3,378**	**2,974**	**2,592**	**2,440**	**2,176**	**325,1**

The next problem set examined is the J90 which contains 480 90-Activity projects. This problem set was not tested as wide as the previous project sets. As a result, the performance of the algorithms is measured with respect to each other's performance.

Table 6-23 GA analysis results of J90 problem set compared with LB

Analysis No	**1000**	**5000**	**25000**	**50000**	**250000**	**1000000**	**Stopping Error**	**No of Opt Sol**
1	6,279	5,666	5,154	4,999	4,636	4,404	4,398	332
2	6,343	5,645	5,073	4,928	4,633	4,365	4,342	334
3	6,359	5,612	5,101	4,984	4,653	4,447	4,444	334
4	6,310	5,672	5,175	4,981	4,656	4,475	4,450	331
5	6,267	5,642	5,118	4,968	4,638	4,441	4,431	332
6	6,279	5,636	5,111	4,956	4,615	4,436	4,423	333
7	6,338	5,577	5,141	4,972	4,686	4,473	4,462	332
8	6,337	5,646	5,131	4,962	4,675	4,454	4,442	328
9	6,313	5,658	5,121	4,909	4,671	4,482	4,477	330
10	6,352	5,630	5,069	4,944	4,685	4,482	4,469	331
Mean	**6,318**	**5,638**	**5,120**	**4,960**	**4,655**	**4,446**	**4,434**	**331,7**

Table 6-24 GASA analysis results of J90 problem set compared with LB

Analysis No	1000	5000	25000	50000	250000	Stopping Error	No of Opt Sol
1	6,302	5,448	3,812	3,395	2,920	2,699	348
2	6,354	5,355	3,837	3,363	2,886	2,665	348
3	6,305	5,414	3,849	3,400	2,930	2,730	347
4	6,309	5,403	3,823	3,381	2,907	2,720	349
5	6,337	5,452	3,859	3,399	2,868	2,678	350
6	6,321	5,418	3,881	3,440	2,903	2,710	348
7	6,303	5,420	3,862	3,404	2,908	2,700	349
8	6,350	5,418	3,811	3,376	2,883	2,691	349
9	6,340	5,443	3,861	3,413	2,885	2,703	349
10	6,344	5,312	3,860	3,391	2,892	2,682	349
Mean	**6,326**	**5,408**	**3,846**	**3,396**	**2,898**	**2,698**	**348,6**

Table 6-25 HGASA analysis results of J90 problem set compared with LB

Analysis No	1000	5000	25000	50000	250000	Stopping Error	No of Opt Sol
1	6,234	5,547	4,266	3,611	2,783	2,601	350
2	6,290	5,510	4,291	3,641	2,853	2,641	349
3	6,239	5,572	4,257	3,644	2,821	2,615	347
4	6,236	5,580	4,277	3,629	2,808	2,640	349
5	6,324	5,532	4,292	3,640	2,784	2,598	349
6	6,263	5,613	4,261	3,661	2,842	2,619	348
7	6,257	5,538	4,321	3,656	2,814	2,608	349
8	6,209	5,550	4,308	3,663	2,813	2,604	349
9	6,296	5,521	4,244	3,644	2,818	2,635	348
10	6,194	5,571	4,274	3,658	2,777	2,631	351
Mean	**6,254**	**5,553**	**4,279**	**3,645**	**2,811**	**2,619**	**348,9**

Table 6-26 PSO analysis results of J90 problem set compared with LB

Analysis No	1000	5000	25000	50000	250000	Stopping Error	No of Opt Sol
1	6,173	5,822	5,518	5,367	5,092	4,927	315
2	6,198	5,828	5,477	5,346	5,064	4,883	314
3	6,205	5,857	5,492	5,369	5,060	4,913	315
4	6,180	5,849	5,538	5,377	5,080	4,906	318
5	6,219	5,840	5,468	5,335	5,102	4,917	315
6	6,159	5,792	5,475	5,352	5,045	4,928	315
7	6,138	5,827	5,488	5,358	5,079	4,958	313
8	6,196	5,844	5,511	5,371	5,090	4,952	314
9	6,172	5,819	5,525	5,390	5,087	4,934	312
10	6,116	5,844	5,505	5,353	5,091	4,919	313
Mean	**6,176**	**5,832**	**5,500**	**5,362**	**5,079**	**4,924**	**314,4**

Performances of the algorithms are very similar to the previous cases. As the problem is a little more difficult than the previous cases the error amount are higher and the number of optimum solutions obtained are slightly lower. GA algorithm is stopped at the end of the 1081000th schedule and both GASA and HGASA are stopped at the end of the 770000th schedule. Due to its slow convergence rate, PSO is stopped at the end of the 650000th schedule. Further iterations of GA is performed in order to analyze the convergence capability of the algorithm. It is seen that the convergence rate excessively decreases after the 250000th schedule. This analysis showed that convergence rate of PSO and GA decreases and there is not any apparent benefit to continue iterations after 250000th schedule.

GASA and HGASA had challenging performances. Similar to the previous analysis GASA and HGASA has more or less the same error range at the end of the 1000th iteration. However, GASA has significantly lower error values than HGASA until the 250000th iteration. After this iteration level, HGASA has faster convergence rate than GASA and catches GASA. Although PSO has the best initial convergence performance at the end of the 1000th run, HGASA, GASA and GA obtains better results at the end of the 5000th schedule. The analysis results in Table 6-23 to Table 6-26 show that GASA has initially significantly fast convergence speed but at the end GASA's convergence speed significantly decreases and converges to almost same error percentage with the HGASA.

Table 6-27 GA analysis results of J90 problem set compared with UB

Analysis No	**1000**	**5000**	**25000**	**50000**	**250000**	**1000000**	**Stopping Error**	**No of Opt Sol**
1	4,105	3,518	3,028	2,882	2,540	2,322	2,316	345
2	4,164	3,499	2,952	2,817	2,537	2,284	2,263	347
3	4,182	3,468	2,979	2,869	2,556	2,362	2,359	346
4	4,134	3,523	3,049	2,867	2,559	2,390	2,366	343
5	4,092	3,495	2,995	2,852	2,541	2,356	2,348	344
6	4,104	3,489	2,988	2,840	2,522	2,351	2,339	346
7	4,162	3,435	3,016	2,855	2,586	2,386	2,376	345
8	4,157	3,497	3,008	2,848	2,577	2,369	2,359	340
9	4,137	3,509	2,998	2,798	2,573	2,396	2,392	340
10	4,175	3,484	2,950	2,831	2,586	2,395	2,383	342
Mean	**4,141**	**3,492**	**2,996**	**2,846**	**2,558**	**2,361**	**2,350**	**343,8**

Analysis results of J90 are also compared with upper bound solutions of J90 problem set. PSO, GA, GASA and HGASA represent satisfactory results and even the error amounts of both GASA and HGASA are smaller than 1%, an improvement in the best known solution set could not be achieved.

Table 6-28 GASA analysis results of J90 problem set compared with UB

Analysis No	**1000**	**5000**	**25000**	**50000**	**250000**	**Stopping Error**	**No of Opt Sol**
1	4,127	3,315	1,772	1,386	0,942	0,737	371
2	4,174	3,227	1,796	1,356	0,913	0,707	366
3	4,131	3,282	1,808	1,390	0,951	0,768	368
4	4,129	3,269	1,783	1,372	0,931	0,757	370
5	4,159	3,315	1,816	1,389	0,894	0,718	371
6	4,142	3,284	1,837	1,427	0,926	0,747	371
7	4,127	3,286	1,819	1,394	0,930	0,738	373
8	4,170	3,284	1,772	1,368	0,909	0,731	370
9	4,165	3,308	1,818	1,402	0,909	0,741	371
10	4,166	3,183	1,817	1,382	0,917	0,722	371
Mean	**4,149**	**3,275**	**1,804**	**1,386**	**0,922**	**0,736**	**370,2**

Table 6-29 HGASA analysis results of J90 problem set compared with UB

Analysis No	**1000**	**5000**	**25000**	**50000**	**250000**	**Stopping Error**	**No of Opt Sol**
1	4,058	3,406	2,198	1,586	0,816	0,648	371
2	4,112	3,373	2,220	1,612	0,881	0,685	372
3	4,065	3,429	2,189	1,615	0,852	0,659	367
4	4,060	3,439	2,209	1,602	0,838	0,683	370
5	4,144	3,393	2,222	1,612	0,818	0,645	368
6	4,084	3,468	2,194	1,631	0,872	0,663	370
7	4,080	3,401	2,248	1,625	0,845	0,655	370
8	4,037	3,410	2,238	1,632	0,844	0,650	371
9	4,120	3,380	2,179	1,615	0,848	0,678	370
10	4,021	3,429	2,205	1,630	0,810	0,675	370
Mean	**4,078**	**3,413**	**2,210**	**1,616**	**0,842**	**0,664**	**369,9**

Table 6-30 PSO analysis results of J90 problem set compared with UB

Analysis No	1000	5000	25000	50000	250000	Stopping Error	No of Opt Sol
1	4,007	3,673	3,387	3,244	2,984	2,829	319
2	4,029	3,678	3,348	3,224	2,959	2,787	317
3	4,034	3,707	3,361	3,245	2,955	2,815	319
4	4,012	3,700	3,406	3,254	2,973	2,808	322
5	4,047	3,689	3,339	3,213	2,993	2,819	319
6	3,992	3,648	3,346	3,230	2,940	2,829	318
7	3,973	3,679	3,357	3,234	2,971	2,858	316
8	4,026	3,697	3,380	3,247	2,983	2,852	317
9	4,004	3,670	3,393	3,265	2,980	2,836	315
10	3,953	3,695	3,376	3,231	2,982	2,821	317
Mean	**4,008**	**3,684**	**3,369**	**3,239**	**2,972**	**2,825**	**317,9**

Final problem set included in the analysis consists of 600 120-Activity projects. As this set has the projects with the highest activity number, this test set is the most difficult one among the others. The analysis results of the comparisons with lower bound are given in Table 6-31 to Table 6-33. GA, GASA and HGASA are stopped at the end of the 1285000th schedule while PSO is stopped at the end of 1100000th schedule. The difference between the schedule numbers is not significant when the slow convergence of the PSO is taken into account.

Table 6-31 GA analysis results of J120 problem set compared with LB

Analysis No	1000	5000	25000	50000	250000	1000000	Stopping Error	No of Opt Sol
1	16,228	14,851	13,714	13,294	12,634	12,134	12,039	140
2	16,259	14,905	13,716	13,330	12,631	12,091	11,994	141
3	16,230	14,946	13,751	13,347	12,615	12,123	12,019	138
4	16,263	14,912	13,779	13,345	12,643	12,084	12,003	133
5	16,269	14,914	13,812	13,374	12,608	12,127	12,010	134
6	16,267	14,924	13,741	13,350	12,641	12,096	12,009	134
7	16,305	14,954	13,783	13,387	12,635	12,152	12,045	135
8	16,136	14,902	13,756	13,351	12,630	12,090	11,978	138
9	16,201	14,906	13,763	13,337	12,611	12,065	11,979	139
10	16,229	14,885	13,740	13,342	12,636	12,095	11,995	140
Mean	**16,239**	**14,910**	**13,755**	**13,346**	**12,628**	**12,106**	**12,007**	**137,2**

Table 6-32 GASA analysis results of J120 problem set compared with LB

Analysis No	1000	5000	25000	50000	250000	1000000	Stopping Error	No of Opt Sol
1	16,289	14,689	11,375	9,684	7,799	7,014	6,906	186
2	16,307	14,618	11,385	9,749	7,858	6,963	6,882	186
3	16,293	14,643	11,389	9,715	7,761	6,926	6,830	186
4	16,214	14,632	11,510	9,736	7,801	6,985	6,892	190
5	16,262	14,703	11,429	9,773	7,782	6,955	6,847	185
6	16,319	14,593	11,413	9,714	7,840	7,026	6,912	188
7	16,255	14,646	11,473	9,732	7,799	6,986	6,888	186
8	16,267	14,668	11,378	9,687	7,795	6,945	6,853	185
9	16,214	14,609	11,429	9,722	7,788	6,993	6,884	187
10	16,262	14,645	11,401	9,719	7,795	6,966	6,868	182
Mean	**16,268**	**14,645**	**11,418**	**9,723**	**7,802**	**6,976**	**6,876**	**186,1**

Table 6-33 HGASA analysis results of J120 problem set compared with LB

Analysis No	1000	5000	25000	50000	250000	1000000	Stopping Error	No of Opt Sol
1	16,136	14,883	13,323	12,167	8,676	7,060	6,946	183
2	16,023	14,861	13,331	12,172	8,699	7,073	6,963	187
3	15,993	14,876	13,297	12,147	8,645	6,995	6,880	189
4	16,122	14,980	13,288	12,139	8,663	7,040	6,901	186
5	16,145	14,915	13,336	12,166	8,675	7,081	6,932	187
6	16,014	14,862	13,262	12,160	8,680	6,982	6,878	190
7	16,118	14,845	13,278	12,088	8,645	7,025	6,875	191
8	16,051	14,903	13,300	12,166	8,678	7,060	6,956	184
9	16,032	14,939	13,269	12,125	8,643	7,058	6,931	189
10	16,096	14,868	13,276	12,148	8,641	7,047	6,891	188
Mean	**16,073**	**14,893**	**13,296**	**12,148**	**8,665**	**7,042**	**6,915**	**187,4**

Table 6-34 PSO analysis results of J120 problem set compared with LB

Analysis No	1000	5000	25000	50000	250000	1000000	Stopping Error	No of Opt Sol
1	15,827	15,208	14,576	14,258	13,661	13,155	13,139	105
2	15,934	15,230	14,556	14,283	13,636	13,144	13,116	107
3	15,797	15,134	14,530	14,257	13,657	13,194	13,165	107
4	15,830	15,166	14,531	14,266	13,646	13,173	13,146	104
5	15,889	15,212	14,550	14,216	13,644	13,188	13,163	101
6	15,905	15,189	14,554	14,247	13,667	13,185	13,170	106
7	15,903	15,192	14,538	14,274	13,629	13,134	13,110	108
8	15,863	15,176	14,506	14,257	13,667	13,142	13,112	107
9	15,910	15,203	14,572	14,288	13,646	13,136	13,097	104
10	15,837	15,208	14,521	14,268	13,660	13,172	13,154	106
Mean	**15,869**	**15,192**	**14,543**	**14,261**	**13,651**	**13,162**	**13,137**	**105,5**

High error percentages of the three algorithms at the end of the 1000^{th} iteration represent the difficulty of the problem set. Similar to the previous problem set, PSO has the best initial convergence results. There is not any significant difference between the successes of the algorithms at the end of the 5000^{th} iteration. However, PSO is the worst algorithm among the four with slight differences. The error percentages are slightly below 15% but the lowest error percentage is obtained by GASA. After this point the high convergence of GASA significantly improves the performance of the algorithm and at the end of the 50000^{th} iteration GASA gives less than 10% of deviation from the lower bound solutions. At this step there is not any significant difference between GA and HGASA which could be the evidence of the convergence capability of the HGASA. However, after the 50000^{th} iteration convergence of HGASA becomes significantly faster than GASA and GA and as a result HGASA almost catches the GASA when the stopping criterion is met. PSO is significantly worse than the other meta-heuristic algorithms. The improvement obtained is very less after the after the 250000^{th} iteration. The convergence capability of PSO is not satisfactory. GA shows satisfactory performance but the algorithm is not as good as GASA and HGASA. There is not a significant difference between GASA and HGASA.

When the analysis results of the PSO, GA, GASA and HGASA are compared with the results obtained by the other researchers given in Table 6-35, it is seen that the four meta-heuristic methods present moderate results. In the comparison, first trials are used out of ten trials. The remaining analyses are performed in order to measure the deviation of the convergences of the algorithms.

Initial convergences of the four meta-heuristic algorithms are very close to each other. PSO presents the best performance at the end of the 1000 schedules. When the first 1000 schedules are compared with the previous studies it is seen that there is significant difference between the four meta-heuristic algorithm and the previous studies. The reason of this can be explained by the activity priority representation since this is the most significant difference of the algorithms.

Table 6-35 Average deviation of the J120 from the optimal makespan

Algorithm	**Reference**	**Max. # schedules**		
		1000	5000	50000
GA-DBH	Debels and Vanhoucke (2005)	34.19	32.34	30.82
GA-hybrid, FBI	Valls et al. (2008)	34.07	32.54	31.24
GAPS	Mendes et al. (2009)	35.87	33.03	31.44
Hybrid scatter	Ranjbar et al. (2009)	35.08	33.24	31.49
GA-forw.-back. -FBI	Alcaraz et al. (2004)	36.53	33.91	31.49
Scatter search-FBI	Debels et al. (2006)	35.22	33.10	31.57
GA-FBI	Valls et al. (2005)	35.39	33.24	31.58
GA, TS-path relinking	Kochetov and Stolyar (2003)	34.74	33.36	32.06
Population based-FBI	Valls et al. (2005)	35.18	34.02	32.81
GA-self-adapting	Hartmann (2002)	37.19	35.39	33.21
Sampling—LFT, FBI	Tormos and Lova (2003b)	35.01	34.41	33.71
GA—activity list	Hartmann (1998)	39.37	36.74	34.03
SA—activity list	Bouleimen and Lecocq (2003)	42.81	37.68	-
TS—activity list	Nonobe and Ibaraki (2002)	40.86	37.88	35.85
GA—priority rule	Hartmann (1998)	39.93	38.49	36.51
Sampling—adaptative	Schirmer and Riesenberg (2000)	39.85	38.70	-
Sampling—LFT	Kolisch (1996b)	39.60	38.75	37.74
Sampling—WCS	Kolisch (1996a, 1996b)	39.65	38.77	-
Sampling—adaptative	Kolisch and Drexl (1996)	41.37	40.45	-
GA—problem space	Leon and Ramamoorthy-1995	42.91	40.69	-
GASA-priority list	This study	45.87	43.85	37.37
GA—random key	Hartmann (1998)	45.82	42.25	38.83
Sampling—LFT	Kolisch (1996b)	42.84	41.84	40.63
HGASA-priority list	This study	45.70	44.12	40.66
Sampling—random	Kolisch (1995)	44.46	43.05	41.44
GA-priority list	This study	45.78	44.07	42.14
PSO-priority list	This study	41,78	46,12	45,04
Sampling—random	Kolisch (1995)	49.25	47.61	45.60

The results of PSO, GA, GASA and HGASA of the J120 problem set are also compared with the upper bounds. The analysis results are represented in Table 6-36 to Table 6-39. The algorithms could not improve any upper bound solution although the mean deviation of project duration is decreased up to 2,3%.

When the convergence characteristics of GASA and HGASA are examined it is seen that the two algorithms still improves its results even after the 1 millionth schedule. It is also expected that if the maximum number of iteration is further increased, the error amount will also decrease. Due to the limitation of computational time, maximum number of schedule is not increased. If the analysis is repeated on a more powerful computer, than the maximum schedule can be increased and better results can be obtained.

Table 6-36 GA analysis results of J120 problem set compared with UB

Analysis No	1000	5000	25000	50000	250000	1000000	Stopping Error	No of Opt Sol
1	11,085	9,769	8,688	8,287	7,661	7,189	7,099	148
2	11,111	9,819	8,682	8,316	7,656	7,145	7,055	148
3	11,082	9,859	8,718	8,336	7,641	7,176	7,079	147
4	11,113	9,826	8,745	8,331	7,668	7,141	7,064	142
5	11,122	9,829	8,775	8,359	7,636	7,182	7,073	141
6	11,119	9,833	8,709	8,339	7,668	7,153	7,070	144
7	11,155	9,867	8,749	8,375	7,660	7,206	7,103	144
8	10,994	9,815	8,725	8,341	7,656	7,144	7,039	148
9	11,058	9,824	8,732	8,326	7,639	7,122	7,041	147
10	11,084	9,801	8,707	8,329	7,664	7,152	7,058	147
Mean	**11,092**	**9,824**	**8,723**	**8,334**	**7,655**	**7,161**	**7,068**	**145,6**

Table 6-37 GASA analysis results of J120 problem set compared with UB

Analysis No	1000	5000	25000	50000	250000	1000000	Stopping Error	No of Opt Sol
1	11,139	9,619	6,476	4,888	3,127	2,393	2,291	219
2	11,155	9,550	6,486	4,947	3,181	2,346	2,270	216
3	11,144	9,574	6,491	4,916	3,092	2,311	2,222	222
4	11,070	9,564	6,601	4,934	3,127	2,363	2,276	223
5	11,113	9,632	6,524	4,970	3,111	2,336	2,236	215
6	11,166	9,530	6,511	4,912	3,163	2,402	2,297	220
7	11,110	9,576	6,570	4,929	3,124	2,366	2,274	216
8	11,122	9,599	6,478	4,886	3,123	2,329	2,242	214
9	11,074	9,541	6,525	4,919	3,114	2,370	2,268	221
10	11,111	9,573	6,498	4,917	3,122	2,346	2,255	217
Mean	**11,120**	**9,576**	**6,516**	**4,922**	**3,128**	**2,356**	**2,263**	**218,3**

Table 6-38 HGASA analysis results of J120 problem set compared with UB

Analysis No	1000	5000	25000	50000	250000	1000000	Stopping Error	No of Opt Sol
1	10,993	9,802	8,319	7,226	3,941	2,431	2,325	218
2	10,884	9,778	8,325	7,230	3,963	2,441	2,340	221
3	10,858	9,794	8,293	7,203	3,913	2,368	2,262	227
4	10,979	9,891	8,285	7,200	3,929	2,414	2,284	219
5	11,003	9,835	8,331	7,223	3,941	2,450	2,310	222
6	10,875	9,781	8,262	7,218	3,946	2,357	2,261	224
7	10,978	9,764	8,277	7,150	3,912	2,395	2,257	225
8	10,912	9,821	8,297	7,224	3,945	2,430	2,333	215
9	10,893	9,851	8,267	7,186	3,911	2,429	2,311	219
10	10,954	9,788	8,275	7,206	3,908	2,418	2,273	221
Mean	**10,933**	**9,810**	**8,293**	**7,207**	**3,931**	**2,413**	**2,296**	**221,1**

Table 6-39 PSO analysis results of J120 problem set compared with UB

Analysis No	**1000**	**5000**	**25000**	**50000**	**250000**	**1000000**	**Stopping Error**	**No of Opt Sol**
1	10,700	10,120	9,521	9,219	8,655	8,178	8,163	108
2	10,801	10,136	9,500	9,243	8,632	8,166	8,139	110
3	10,669	10,045	9,475	9,217	8,651	8,214	8,187	110
4	10,704	10,077	9,477	9,227	8,641	8,194	8,169	107
5	10,762	10,123	9,497	9,179	8,641	8,211	8,187	104
6	10,776	10,100	9,499	9,210	8,663	8,206	8,193	109
7	10,770	10,100	9,484	9,234	8,626	8,159	8,135	111
8	10,735	10,088	9,455	9,219	8,660	8,164	8,136	110
9	10,777	10,112	9,517	9,248	8,643	8,160	8,122	107
10	10,710	10,115	9,467	9,229	8,656	8,193	8,176	109
Mean	**10,740**	**10,101**	**9,489**	**9,223**	**8,647**	**8,184**	**8,161**	**108,5**

The difference of the convergence ability of the algorithms becomes more significant when upper bounds are used for the comparison. GASA and HGASA converged into upper bounds twice much of PSO had converged. GA is between PSO and GASA and HGASA.

6.4 Conclusion

PSO, GA, GASA and HGASA showed fabulous performance and had successfully entered into the list. Although the algorithms could not take place in the higher positions, it can be easily concluded that by simple modifications and experimental design, convergence capability of the algorithms can be improved further and the algorithms can take better positions in the list.

The difference between the performances of GA, GASA and HGASA can be explained as the affect of simulated annealing which prevents performing harmful mutations. The close error percentage values at the end of the 1000th iteration can be seen as the evidence of this inference. In the beginning, the temperature is high enough to allow harmful mutations to occur but in the later analysis the temperature decreases and the probability of acceptance of harmful mutations decreases. Thus when harmful mutations are accepted in GA, they are probably rejected by GASA and HGASA.

The initial values of PSO are very close to GA based algorithms. Even in J60 and J120 problem sets, initial convergence of PSO is better. However, due to its slow convergence at further schedules PSO presents worst results among the four algorithms. The similar initial error values of the four algorithms represents that the success of the algorithms mainly depends on the activity priority representation.

GASA and HGASA showed different convergence characteristics. HGASA performs a detailed local search on the current best individual in order to improve the best value. While GASA do not perform a local search and continuously seeks to improve the population by continuously performing crossover and mutation operations. The reason of slow convergence of HGASA in the beginning can be explained as the best individuals are not good quality genes in which they can be improved easily. In addition to this, initial local search may cause harmful mutations to be accepted when the heat is high. However, by continuous crossover and mutation operations the overall quality of the population is improved and the initial best gene is overtaken by high quality genes. For this reason, convergence of HGASA requires more iteration than GASA.

PSO, GA, GASA and HGASA represent significantly better performance than the previous algorithms. The reason of this can be explained by several factors. First of all the populations are generated randomly by avoiding any a-priori information. Generating the seeds by taking into account float times, number of successors and resource usages may lead to generation of poor quality seeds which may not cover the search space properly. The second factor can be explained as the gene or position representation. In the previous studies obtained from the literature, genes are sequenced according to the priorities and the activities' positions are shifted by the crossover and mutation operators. However, in this algorithm the activities' positions are kept constant and a unique priority is assigned to each activity. The priorities are checked for the CPM logic and the necessary corrections are performed. As this is the case, when crossover operation is implemented, the genes are not shuffled harshly. The priority representation indexed by activity order prevents excessive shuffling of activity priorities and encourages a systematic search through the solution space.

Activity priority representation converges to optimum schedule by several combinations which are shown by the 8-Activity resource constrained project of Toklu. This property significantly increases the number of combinations that converges to global optimum solution and increases the convergence speed.

With the activity priority representation and adapted crossover and mutation operators, significant improvements are obtained which makes the meta-heuristic algorithms a good candidate for implementing the algorithms for the solution of SRCPSP.

CHAPTER 7

MULTI-MODE RESOURCE CONSTRAINT SCHEDULING PROBLEM

Multi mode resource constraint scheduling problem (MRCPSP) deals with project scheduling problem in which activities have more than one execution mode in which available resources are restricted. In this chapter MRCPSP type problems are analyzed by using PSO, GA, GASA and HGASA meta-heuristic algorithms. Data sets used for the evaluation of the performance of the methods are obtained from http://129.187.106.231/psplib/main.html. Among the MRCPSP type problems J10, J12, J14, J16, J18, J20, J30, R1, R3, R4 and R5 are used. JXX series involves two limited renewable and two limited non-renewable resources where XX represents the activity number of the project. RX series consist of 16 activity project with two non-renewable limited resource and with X renewable resource. In the next subchapter the genetic algorithm based meta-heuristic algorithms and the solution algorithm of MRCPSP are briefly described.

7.1 Genetic Representation of MRCPSP Problem

Similar to the SRCPSP type problems, in MRCPSP type problems, gene represents the priority of the activity. In addition to the activity priority, construction mode of the activity is also represented in the genes of the individual. Activity priority representation is kept same with the SRCPSP type problems. For the representation of the construction mode of the activities integer coding is preferred. The reason of this representation is to limit the search space of the problem.

If binary coding is represented, necessary mutations for the convergence of optimal solution would increase. If the problem was handled by unlimited resources the enlarged search space would not be a serious problem. However by considering the activity priorities for the limited resources the number of combinations is enlarged enormously.

Integer representation limits the search space, but it does not have the advantages of the binary coding. Binary coding may end up with new construction mode with the crossover operator while integer genetic representation does not have this opportunity. In order to obtain acceptable solutions in minimum number of iteration integer type representation is preferred.

Genetic representation for MRCPSP consists of generation of population, crossover, mutation and natural selection. In the following sections the operators of genetic algorithm for MRCPSP is explained.

Genes of the individuals represents both the priority of the activity and construction mode of the activity in MRCPSP type problems. Activity priority is exactly the same with the SRCPSP type problems. For the representation of construction mode of the activities, integer representation of construction modes is preferred instead of binary encoding. The integer representation prevents the widening of the search space. The number of mutations necessary for the binary encoding may be much higher than the integer encoding. When the search space for the activity priorities and activity modes are considered at the same time it would be impossible to obtain reasonable solutions in small number of schedules. However, by preferring integer type of representation, some advantages of binary encoding are missed.

The gene of an 8-Activity project for the MRCPSP type problem is shown in Figure 7.1. Minimum value of the construction modes is 1 and the maximum value is the number of construction modes assigned for that activity. The genes representing construction modes are initially generated by assigning randomly generated integer numbers between 1 and number of construction modes assigned for that activity.

Activity Priorities

1	5	3	4	2	6	7	8

Construction Modes

1	2	2	1	1	3	1	3

Figure 7.1 Gene representation for MRCPSP type problems

Mutation operator may alter activity priority or one of the construction modes. In order to decide which representation to mutate an integer number between 1 and 2 is generated for the decision. If the random number is 1 activity priority is mutated, if the random number is 2 then construction modes are mutated.

For the mutation of the construction mode of an activity, an integer random number between 1 and the number of activity in the project is generated. The generated random number indicates which activity to be mutated. Another integer random number between 1 and 1 less of the number of construction mode of that activity is generated. The generated random number and the current value assigned for the construction mode is summed. If the summation is bigger than the number of construction modes, than number of construction modes is subtracted from the summation. Obtained number is assigned for the construction mode. The aim of generating random number which is 1 less of the number of construction modes is to prevent obtaining same construction mode. The mutation operator for the activity priority is the same with the mutation operator for SRCPSP type problem.

Crossover of the genes for the activity priority and construction modes is performed simultaneously. Crossover of the genes for the activity priority is the same with the crossover operator of the SRCPSP type problems. The crossover operator of the construction modes is performed with the same parents and by using the same crossover point of the activity priority. The crossover operator of construction modes is shown in Figure 7.2.

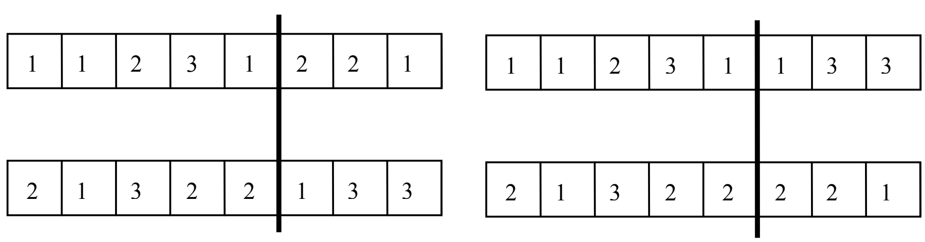

Figure 7.2 Crossover operator of the construction modes

Implementation of PSO for MRCPSP is similar to the implementation of SRCPSP. Position of a particle represents activity priority and crashing mode. Similar to GA based meta-heuristic algorithms there are two separate representations for the activity priority and crashing mode.

Activity priority representation of MRCPSP is exactly the same with the representation in SRCPSP. Similarly, crashing mode representation is exactly the same with the representation in TCT problem. Consequently, for each activity there are two position and velocity values. Since in the problem sets there are three construction modes for each activity, range for the position representing the crashing mode is assigned as three. Maximum velocity of the crashing alternatives is assigned as 1.1.

Alteration of construction modes in MRCPSP does not have any affect on activity priorities. For this reason, there is no need to check for the CPM logical relations after a change in the construction modes. However, alternating the construction modes affects the resource usage of that activity. New construction mode combination may violate the non-renewable resource limits.

Generating proper individuals which do not violate the non-renewable resource is very difficult for some of the projects. Very small portion of the execution mode combinations satisfies the restrictions. As this is the case, new activity execution modes may not satisfy the restrictions which are generated by mutation and crossover operators. In order to immediately reach to a proper combination, an initial pool of activity execution mode combinations is generated when the population is initialized. One of the proper execution modes in the pool is randomly selected and assigned to the violating individual.

If there is an improvement in the current best, one of the randomly selected execution mode combinations in the pool is replaced by the current best gene combination. This replacement aims to improve the quality of the execution modes in the pool. If the pool is kept constant during the optimization procedure, it will decrease the overall quality of the population if there is a non-renewable resource violation especially at the later iterations.

Implemented meta-heuristic algorithms; GA, GASA and HGASA, are based on mutation, crossover and natural selection operators, while PSO is based on the past experience of the individuals and the best individual of the population. The pseudo-codes of the algorithms are given below.

begin
Set the population size, Ps to 1,5* no of activities of the Project Ns, the crossover probability, Pc is set as 0.4, the mutation probability, Pm is set as 0.1 and the stopping condition is Nc^3 model generation.
Generate population randomly
Check for the Network logical restrictions
Model generation counter i is set as 0
While i < Nc^3:
Select individuals from P for mutation with probability Pm.
Generate a random number 1,2 to decide on priority mutation or construction mode mutation
Perform mutation
    Check for the Network logical restrictions if priority mutation
    Check for resource usage if construction mode mutation.
Select chromosomes from P for crossover with probability Pc.
Randomly match the individuals selected for crossover
Perform crossover
    Check for the Network logical restrictions
    Check for the resource usage
Assign probability of survival with the inverse of project completion duration, 1/Pd
Randomly select the individuals by elitist roulette wheel for natural selection
Terminate equal number individuals produced in crossover
i = i + 1
end

The operators mentioned above form GA for MRCPSP. In order to improve the solution capability of GA, mutation operator is modified by SA and the resulted method is called GASA. Modified mutation operator accepts or rejects a mutation based on the mutation's results and the temperature. If the mutation is beneficial, it is always accepted. On the other hand, harmful mutations are accepted or rejected based on the temperature and the amount of elongation of project duration. In the earlier iterations the probability of acceptance of harmful mutations are higher. The pseudo-code of GASA is given as;

begin
Set the population size, Ps equal to1,5* no of activities of the Project Ns, the crossover probability, Pc is set as 0.3, the mutation probability, Pm is set as 0.4 and the stopping condition is Nc^3 model generation, Boltzmann Constant is set to the project completion duration with unlimited resources.
Set temperature T initially to Nc^3
Generate population randomly
Check for the Network logical restrictions
Model generation counter i is set as 0
While $i < Nc^3$:
Select individuals from P for mutation with probability P_m.
Generate a random number 1,2 to decide on priority mutation or construction mode mutation
Perform mutation
    Check for the Network logical restrictions if priority mutation
    Check for resource usage if construction mode mutation.
Check for the acceptance criteria and accept if $Rn < \exp(([f_i - f_i']/ f_i)*BC/T)$ where Rn is a randomly generated number between 0 and 1, fi is the project duration before mutation, f_i' is the project duration after the mutation.
Select chromosomes from P for crossover with probability Pc.
Randomly match the individuals selected for crossover
Perform crossover
    Check for the Network logical restrictions
    Check for resource usage
Assign probability of survival with the inverse of project completion duration, 1/Pd
Randomly select the individuals by roulette wheel for natural selection
Terminate equal number of individuals produced in crossover
i = i + 1
T = T -1
end

Both GA and GASA do not have a local search capability. In order to improve the GASA further a local search operator is embedded into GASA and hybrid meta-heuristic algorithm based on GA and SA (HGASA) is formed. With embedded local search operator the algorithm seeks for better neighbors of the best gene and randomly selected genes. Local search operator is executed after several generation of GA. In order to limit the computational burden, whole population is not exposed to local search. Best gene is exposed to local search in order to obtain better gene representations. Local search of only best gene may cause being stuck into local minima if best gene is not close to global optima. For this reason, some of the randomly selected genes are exposed to local search. The local search gradually improves the population's overall gene quality and increases the probability of producing better genes at the end of crossover operator. The pseudo-code of HGASA is given as;

```
begin
Set the population size, Ps to 1,5*no of activities of the Project Ns, the crossover probability, Pc is set as 0.3, the mutation probability, Pm is set as 0.4 and the stopping condition is Nc^(3/2) model generation, Boltzmann Constant is set to the project completion duration with unlimited resources.
Set temperature T initially to Nc^(3/2), Local search period, Ls is set as 5.
Local search amount, La is set as 4*Nc, and searched individual number Si, is set as 4.
Generate population randomly
Check for the Network logical restrictions
Model generation counter i is set as 0
While i < Nc^(3/2):
Select individuals from P for mutation with probability Pm.
Generate a random number 1,2 to decide on priority mutation or construction mode mutation
Perform mutation
        Check for the Network logical restrictions if priority mutation
        Check for resource usage if construction mode mutation.
Check for the acceptance criteria and accept if Rn < exp(([fi – fi']/ fi)*BC/T) where Rn is a randomly generated number between 0 and 1, fi is the project duration before mutation, fi' is the project duration after the mutation
Check for the Network logical restrictions
Select chromosomes from P for crossover with probability Pc.
Randomly match the individuals selected for crossover
Perform crossover
        Check for the Network logical restrictions
        Check for resource usage
Check for the Network logical restrictions
Assign probability of survival with the inverse of project completion duration, 1/Pd
Randomly select the individuals by roulette wheel for natural selection
Terminate equal number of individuals produced in crossover
Check if hybrid Local search condition satisfied (i % Ls = 0)
        if true
        j = 0
        Select current best individual and randomly Si - 1 individuals
        While j < La:
        Perform mutation
        Check for the acceptance criteria and accept mutation if Rn < exp((fi – fi')*CB)*BC/T)
        where CB is the project completion duration of the current best individual
        Check for the Network logical restrictions
        j = j  + 1
        end
i = i + 1
T = T -1
end
```

Local search continuously mutates the selected gene. The decision of the acceptance of the mutations is taken based on SA. The decision for the acceptance of the mutations has vital importance since too easily acceptance of harmful mutations may cause termination of good genes and too strict acceptance criteria may prevent escaping from local optima.

Finally pseudo-code of PSO is given below.

begin
Set the population size, Ps to 1,5* no of activities of the Project Ns, the stopping condition is Nc^3 model generation.
Generate particles randomly
Check for the Network logical restrictions
Model generation counter i is set as 0
Evaluate particles
While i < Nc^3:
Evaluate particles
Update Population best and particles' overall best
Compute velocities for the activity priorities and crashing modes
Update positions
    Check for the Network logical restrictions if priority mutation
    Check for resource usage if construction mode mutation.
Evaluate particles

i = i + 1
end

7.2 Analysis Results

Problem sets JXX and RX are solved by the algorithms PSO, GA, GASA and HGASA adopted for multi-mode projects. The analysis results are compared with the best solutions obtained from the literature. The results are tabulated starting with the J10 problem set to R5 problem set. The analysis results are grouped according to the meta-heuristic solution algorithm. Mean of the 10 solution of the each problem set is provided in the tables from Table 7-1 to Table 7-3. The columns of the table represent the average error of the solution at the end of the corresponding iteration number. The last two columns represent average number of optimum solutions obtained and the number of projects at that problem set respectively.

Table 7-1 Analysis results of GA

Problem Set	1000	5000	25000	50000	250000	1000000	Overall	No of Opt	No of Projects
J10	12,478	5,151	2,194	1,518			0,961	472,4	536
J12	15,857	7,111	3,516	2,587			1,341	448,4	547
J14	19,475	9,509	4,849	3,700	1,989		1,683	407,2	551
J16	22,918	12,133	6,425	4,983	2,615		1,991	382,6	550
J18	25,313	13,884	7,234	5,590	3,028		2,126	371,4	552
J20	27,727	16,261	8,624	6,701	4,024		2,848	334,2	554
J30	36,714	25,759	16,754	14,652	11,094	8,678	8,139	198,9	542
R1	21,694	10,821	5,464	4,188	2,197		1,622	417,2	553
R3	22,579	11,802	6,069	4,656	2,418		1,826	393,5	557
R4	23,209	12,564	6,637	4,912	2,648		2,211	367	552
R5	23,768	13,092	6,935	5,337	2,923		2,257	358	546

Table 7-2 Analysis results of GASA

Problem Set	**1000**	**5000**	**25000**	**50000**	**250000**	**1000000**	**Overall**	**No of Opt**	**No of Projects**
J10	10,552	3,854	1,349	0,836			0,307	511,5	536
J12	14,437	6,481	2,647	1,718	0,661		0,550	499	547
J14	18,076	9,072	4,233	2,963	1,327		0,890	459,7	551
J16	21,854	11,672	5,443	3,948	1,898		1,203	428,9	545
J18	24,764	13,810	6,573	4,709	2,223		1,346	420,9	552
J20	27,405	16,259	7,966	5,704	2,749		2,103	372,9	554
J30	36,412	26,077	17,538	14,503	7,701	3,812	3,354	315,3	542
R1	20,813	10,854	5,131	3,620	1,629		1,000	459,9	553
R3	21,849	10,740	4,184	2,681	0,995		0,788	460,8	557
R4	22,432	11,889	5,593	4,048	1,915		1,287	427,8	552
R5	23,255	12,642	5,808	4,065	1,824		1,189	430,1	546

Table 7-3 Analysis results of HGASA

Problem Set	**1000**	**5000**	**25000**	**50000**	**250000**	**Overall**	**No of Opt Sol**	**No of Projects**
J10	11,304	4,146	1,260	0,727		0,476	499,8	536
J12	15,995	7,449	2,952	1,843		0,840	476,5	547
J14	19,971	10,541	4,886	3,368		1,459	418,5	551
J16	24,094	13,783	6,979	4,980	2,207	1,960	378,8	545
J18	26,795	16,425	8,434	6,054	2,668	2,194	365	552
J20	29,663	19,501	10,676	7,715	3,374	2,519	351,2	554
J30	38,235	30,347	22,214	18,978	11,476	6,741	223,5	542
R1	23,083	12,717	6,106	4,318	1,876	1,687	410,2	553
R3	24,205	13,914	6,762	4,781	1,994	1,757	396,3	557
R4	24,925	14,377	7,198	5,126	2,262	1,996	379,6	552
R5	25,440	15,264	7,655	5,359	2,223	1,986	376	546

Table 7-4 Analysis results of PSO

Problem Set	**1000**	**5000**	**25000**	**50000**	**250000**	**Overall**	**No of Opt Sol**	**No of Projects**
J10	31,683	31,681	31,681	31,680		31,680	13,6	536
J12	33,368	33,365	33,364	33,364		33,364	7,1	547
J14	35,773	35,761	35,757	35,757	35,757	35,757	3,5	551
J16	37,283	37,273	37,265	37,265	37,265	37,265	2,8	545
J18	38,812	38,791	38,774	38,771	38,768	38,768	1,6	552
J20	41,437	41,424	41,411	41,372	41,369	41,357	1,1	554
J30	44,141	44,076	44,029	44,006	43,963	43,939	0,0	542
R1	36,388	36,379	36,378	36,378	36,378	36,378	3,2	553
R3	38,276	38,250	38,241	38,239	38,237	38,235	2,6	557
R4	38,496	38,474	38,466	38,464	38,459	38,458	3,2	552
R5	39,150	39,125	39,114	39,113	39,111	39,111	1,8	546

When the analysis results are examined it is seen that error values are significantly high especially in the earlier iteration stages. The high error values in the earlier stages illustrate the difficulty of the problem. In the early stages there is not any significant difference between the convergence capabilities of the algorithms; however especially after the 25000th iteration mean error value of GASA becomes significantly lower than PSO, GA and HGASA.

GA and HGASA end up with similar error values when the stopping criteria are met. However, GA presents slightly worse results than HGASA presents. There was a significant difference between GA and HGASA in SRCPSP. However, there is still a difference but not that much significant. The local search operator does not improve the results as expected. Especially, in the initial evaluations GA and HGASA mean error values are very close to each other. Only after the 50000th iteration, HGASA can present better results than GA. The difference becomes significant especially in J20 and J30 problem sets. The late improvement of local search can be explained as; in the initial case, when the temperature of the environment is high, harmful mutations are possible to be accepted. During the random walk session HGASA continuously mutates the individuals in order to obtain better scheduling alternatives. Especially in the initial stages acceptance of a detrimental mutation is very high and almost all detrimental mutations are accepted. This situation requires more beneficial mutations to improve the current best. On the other hand, if the initial temperature is decreased than the probability of getting stuck into local minima will increase.

It is seen that the difficulty of MMRCPS is very high that the initial error values of the algorithms are too high. Because of this difficulty, the probability of a mutation to be beneficial is very low and the initial random walk sessions are prone to end up with detrimental mutations. Detrimental mutations are also prone to GASA and GA but, the two meta-heuristic algorithms do not include a random walk search algorithm. So that GA and GASA do not have as many detrimental mutations as HGASA have. After the environment cools the amount of detrimental mutations lowers and the mean error of HGASA catches GA in the end.

PSO present the worst results. In this case, the initial convergence results of PSO are not as successful as SRCPSP. In addition to this, even this initial mean error values are not improved. It can be inferred that with this position and crashing alternative representation, PSO is not suitable for MMRCPS.

When the stopping criterion is met, maximum mean error value of GASA is %3,354 which is an acceptable value for this much difficult problem. On the other hand, the algorithms could not improve any optimum duration. Accept for the J30 problem set, GASA obtained approximately 350 global optimums out of 550 test problems and is the most successful meta-heuristic algorithm.

Table 7-5 Convergences of the algorithms when stopping criteria is met

Problem Set	**GA**		**GASA**		**HGASA**		**PSO**	
	Av Dev	Per Op Found	Av Dev	Per Op Found	Av Dev	Per Op Found	Av Dev	Per Op Found
J10	0,96	88,13	0,31	95,43	0,48	93,25	31,68	2,54
J12	1,34	81,97	0,55	91,22	0,84	87,11	33,36	1,30
J14	1,68	73,90	0,89	83,43	1,46	75,95	35,76	0,64
J16	1,99	69,56	1,20	78,70	1,96	69,50	37,27	0,51
J18	2,13	67,28	1,35	76,25	2,19	66,12	38,77	0,29
J20	2,85	60,32	2,10	67,31	2,52	63,39	41,36	0,20
J30	8,14	36,70	3,35	58,17	6,74	41,24	43,94	0,00
R1	1,62	75,44	1,00	83,16	1,69	74,18	36,38	0,58
R3	1,83	70,65	0,79	82,73	1,76	71,15	38,24	0,47
R4	2,21	66,49	1,29	77,50	2,00	68,77	38,46	0,58
R5	2,26	65,57	1,19	78,77	1,99	68,86	39,11	0,33

In Table 7-5 deviation of the project durations when the stopping criteria is met is given. Apparently, GASA is the most successful algorithm among the four heuristics. When the results are compared with the previous studies it is seen that GASA is not successful in J10 and J12 problem sets as it can not provide better results than any of the distributed results. In J14, J16, J18 and J20 problem sets GASA provides better results than Sprecher and Drexl (1998) and Hartmann (2001).

CHAPTER 8

RESOURCE CONSTRAINED TIME COST TRADE-OFF PROBLEM

The final problem type analyzed in this thesis is the optimization of resource constrained time cost trade off problem (RCTCT). RCTCT is similar to the TCT analysis, in which each activity may have more than one execution mode with different completion duration and cost. However, in RCTCT type problem each execution mode's resource requirements are also assigned and the maximum amounts of these resources are limited. The aim of RCTCT optimization is the minimization of the total project cost without overriding the resource restrictions.

The meta-heuristic algorithms GA, GASA and HGASA are implemented for the solution of RCTCT.

8.1 Genetic Representation of RCTCT Problem

Genetic representation preferred for the solution of RCTCT problem is exactly the same with the MRCPSP problem. Gene of an individual represents both the priorities of the activities for the resource usage and the activity execution mode. The detailed information about the genetic representation and solution algorithm can be obtained from Chapter 7. The only difference between MRCPSP and RCTCT is that the objective function is minimization of total project cost instead of minimization of project duration.

8.2 Problem sets for RCTCT Problem

In the literature there was not a problem set available for the RCTCT set. For this reason, random problems sets are generated. For the base network 120-activity projects obtained from PSPLIB are used. Readily generated multi-mode projects are not preferred because the number of activities is less and the number of execution modes is constant for each activity. In addition to this, some of the crashing alternatives do not satisfy the renewable resource requirements.

Additional activity execution modes are added to the project by assigned number. For each activity a random number between 1 and 4 is generated and additional execution modes equal to the random number generated for that activity are created. Consequently, each activity would have execution modes between 2 and 5. The original duration of the activities are assumed to be its full crashed duration of that activity. The reason of this is the too short original activity durations. The original durations are between 1 and 10 days which are not possible to crash.

For each execution mode added to the original problem another random number between 1 and 3 is generated. This random number is the duration difference between the next crashing option and itself. In other words, the original duration is assigned as the shortest possible duration for the completion of the activity. Generated random numbers are added to the duration of the previous crashing option in order to compute the current crashing option itself. As a result additional execution modes will take longer to complete the activity than the original duration.

Resource requirement of the activity is computed by multiplying the duration of the activity and the required amount of that activity. Man-days for each resource type are computed and they are tried to be kept constant for each mode. If there is an increase in the amount of man-days for a certain resource type because of rounding errors, the resource requirement is round-off in order to preserve this. However, if the resource requirement becomes zero after the rounding off, than its original value is preserved.

By crashing the activity durations it is assumed that the efficiency of the workers would decrease when compared with the efficiency of the workers in activity completion in normal duration. It is assumed that the labors are worked over-time in order to prevent additional resource for the decreased efficiency and the job is completed with same man-days. Crashing of normal activity durations comes at an additional cost. This additional cost is taken as $200 for each crashed day. The total cost of the activity execution modes are computed by the summation of all resource costs and the overtime cost of the activity. Labor costs are computed by the summation of the multiplication the unit costs of the resources with the man-day requirement for that resource. Unit costs of the resources are assumed to be; $50/day for R1, $55/day for R2, $60/day for R3 and $65/day for R4.

8.3 Analysis results

The RCTCT problem is analyzed by PSO, GA, GASA and HGASA by the 600 projects for $750/day, $1500/day, $2250/day and $3000/day overhead costs.

At the end of the RCTCT optimization, the final schedule is further worked on in order to obtain better resource distribution. The summation of the square of the daily resource usage is tried to be minimized by randomly delaying the activities without prolonging the original project completion date and overriding of the resource limits.

Table 8-1 Analysis results of RCTCT problem

Algorithm	**Overhead**	**1000**	**5000**	**25000**	**50000**	**250000**	**500000**	**Duration**
GA		591298	588894	584962	582396	578332	577607	141,3
GASA		590535	588421	585702	583329	562737	543854	133,6
HGASA		594039	591703	589630	588286	576041	564692	138,8
PSO	750	612684	608773	602419	599358	592188	587205	143,0
GA		699698	696300	691462	689061	684822	683649	137,1
GASA		700322	697183	693000	689668	666289	645316	128,5
HGASA		699754	696273	693328	691098	677253	665717	132,5
PSO	1500	708923	703744	696108	692527	684164	677760	138,1
GA		806814	802191	796140	793120	787504	785899	135,4
GASA		807250	802880	797093	792177	765409	742414	126,6
HGASA		807033	802116	797744	794568	777818	765545	130,7
PSO	2250	917033	912116	907744	904568	887818	875545	150,7
GA		913291	906997	899300	895674	889215	887674	134,6
GASA		913520	907681	900538	894159	863118	838058	125,7
HGASA		913431	907567	901460	897218	876966	863940	129,6
PSO	3000	1023431	1017567	1011460	1007218	986966	973940	149,6

The analysis results are shown in Table 8-1. Cost values are the average of the 600 project at the corresponding iteration value. Iterations are stopped when the project schedule number reaches 500000. Duration value is the average of the durations of the project with the least total project cost. It is seen that when the overhead cost increases, average project duration decreases.

GASA is the most successful meta-heuristic algorithm when the methods are compared. HGASA gives slightly worse results than GASA; however HGASA is better than GA and PSO. PSO is the worst method among the four meta-heuristic algorithms.

The success of GASA shows that, SA significantly improves the convergence ability of GA. As a result of this, GASA obtained the best solutions among the four meta-heuristic algorithms. Local search algorithm of HGASA slows down the convergence of the algorithm. Analysis results of MRCPSP and RCTCTP show that HGASA improves its results at later schedules. HGASA requires more iterations than GASA in order to obtain same quality results. As a result of this, HGASA gives worse results when the stopping criterion is met.

GA obtained worse results than GASA and HGASA. This shows the benefit of SA. Both HGASA and GASA represent better results in any of the projects. The results of PSO show that, activity priority representation is not a suitable algorithm for PSO to improve its initial convergence. It is known that PSO is a fast converging meta-heuristic algorithm. The obtained results do not reflect the characteristic of PSO. On the other hand, obtained solutions by PSO is also satisfactory enough for the execution.

The particles in the population of PSO are directed towards the population's current best. The activity priorities are altered in a way which will make them as same as with the population's current best's activity priorities. The process is performed step by step in order to search for better activity priority combinations. However, due to the CPM logical relationships activity priorities can not be altered freely. The altered activity priorities probably will not satisfy the CPM logical relationships and they are corrected accordingly and the position vector representing activity priority list is updated.

Thus the obtained activity priorities will not be able to follow the path towards the population current best. As a result of this, the particles will make a random search through their region inefficiently.

The 120-activity project problem set is significantly a difficult problem set. Since the problem set contains activities with multi execution modes and the project has resource constraints to satisfy. The possible combinations are too high that the initial population may not be rich enough to cover all possible combinations. Even the 120-activity TCT analysis with unlimited resources is a difficult problem, as a result the problem set contains very challenging problem set. For this reason, it is normal that the convergence characteristics of the meta-heuristic algorithms change significantly. The analysis results show that the algorithms should be improved and the test problem should be worked on it.

The 120-activity RCTCTP test set is the most difficult problem set analyzed in this thesis. Number of possible combinations is extremely high which can not be compared with any of the test sample.

When the stopping criterion of RCTCTP is met, the obtained schedules are analyzed in order to perform resource leveling. In this case, it is not possible to compute floats of the activity since the activity start times are not only determined by CPM relationships but also by resource availabilities as well. As this is the case, each activity is shifted one day and the project is rescheduled. If there is not any override of resource restriction and no elongation of project duration, the shift is accepted. Otherwise it is rejected.

All of the activities are sequentially examined. If there is any accepted shift, than activities are randomly shifted in which the activity to be shifted is determined randomly. This process is repeated 25000 times. At the end of the process, it is seen that there is not any improvement in the resource profiles of the activities.

This is caused by the fact that, the resource profiles are too strict that delay of an activity probably causes an increase in project duration or override of a resource restriction. In addition to this, since the schedules are near-optimum schedules which is the almost shortest project durations where the project can be executed with the available resources. For this reason, there are not significant fluctuations in resource profile.

CHAPTER 9

CONCLUSION

In this thesis, it is aimed to obtain optimum or near-optimum solutions for the planning optimization problems including; time cost trade-off, resource leveling and resource constrained project scheduling problems by meta-heuristic algorithms. The reason of preferring the meta-heuristic algorithms is that some of the planning optimization problems, such as multi-mode resource constrained scheduling and resource constrained time cost trade-off analysis can not be solved by linear programming or other analytical methods, especially for large projects.

In order to develop and improve convergence capability of meta-heuristic algorithms in the beginning simple planning problems are solved such as TCT analysis and resource leveling. Each problem type was a good candidate to detect the weak points of the algorithms and provide a road map to improve the algorithm.

In the first problem type, TCT with unlimited resources is analyzed. The scheduling software developed for this type of problem is capable of handling all of the four logical relationships. In addition to this, delays and lags can be defined as well. Four different calendars can be assigned to the activities which are the most suitable calendar types for the industry.

In the analysis it is shown that the HGASA, GMASA, GASAVNS are capable of obtaining global optima for the middle sized projects. However, if near optimum solutions are acceptable than ACO, PSO and GASA are also capable of providing satisfactory results. The solution of TCT problem with unlimited resource problem type can be adapted to construction sector for minimizing the total project cost if there is not any difficulty in obtaining and accommodating any kind of labor. The developed meta-heuristic algorithms are good candidate to reduce the total cost of the construction projects.

In the literature, the largest project whose global optimum is obtained by meta-heuristic algorithm was only 18-activity project. In this thesis, 63-activity project is analyzed by meta-heuristic algorithms and three of them were able to obtain the global optima of the project. This is a sound improvement for the capability of meta-heuristic algorithms.

HGAQSA present the most successful results in the analysis of TCT problem. The algorithm always converged into optimum or near-optimum solutions. As a result of this, HGAQSA can be implemented for the solution of TCT problems without any doubt. In addition to this, PSO present near-optimum solution in the early stages of the analysis. If optimum solution is not mandatory which can be the case if there are uncertainties in the planning data, PSO can be preferred in order to shorten analysis duration. Optimum solution of TCT presents the minimum total project cost. As a result of this, optimum solution of TCT problems has significant importance in construction and manufacturing sectors.

Second problem type was resource leveling in which there was no limit on the maximum resource amounts. This problem is also an important scheduling problem in construction sector, which aims to minimize the fluctuations in the daily resource demand during the project. The fluctuations in the resource demand causes idle resources and very often hiring and firing labors which reduces the unit production.

For resource leveling, the meta-heuristic algorithms are tested by using the test problems obtained from the literature. In one of the test problems better leveling of resources is obtained when compared with the results in the literature and same schedule is obtained in the remaining test problems. The test problems showed that meta-heuristic algorithms; GASA, HGAQSA, GMASA and GASAVNS are capable of improving the resource demand during the construction and provides better resource curve suitable for an efficient and productive construction.

Similar to TCT problems, HGAQSA obtained the most successful results in shortest time. By considering this fact, in the analysis of large projects HGAQSA would be more suitable than the other algorithms.

Optimum solution of resource leveling reduces the fluctuations in resource demand. As a result of this, more stable resource curves are obtained which reduces the amount of idle labor and machinery during the project execution. Consequently, additional costs occurring from idle resources would be prevented by the optimum or near-optimum solution of resource leveling.

Third problem type analyzed is the resource constrained scheduling problem. In this case three genetic algorithm based methods are preferred in the analysis; GA, GASA and HGASA. In addition to this, the problem is also analyzed by PSO. RCPSP is commonly faced in manufacture and construction sectors. The solution of the problems aims to minimize the duration of the production or construction as short as possible without overriding the resource limitations and logical relations. Especially in international constructions or in some local regions there might be some limitations in providing some of the skilled labors such as crane operators, welders or carpenters. In such a case obtaining additional skilled labor can be impossible or very expensive. Because of this, maximum numbers of labors are limited to a certain number and the project duration is allowed to be elongate. SRCPSP aims to minimize the elongation of project duration due to resource limitations.

Randomly generated test problems are used for the test of the generated meta-heuristic algorithms. The algorithms could not improve the currently obtained best solutions. However, test results showed that PSO, GA, GASA and HGASA are suitable algorithms for the near optimum solutions of SRCPSP both for construction and manufacture sectors. However, GASA is the most suitable for the analysis of the SRCPSP.

Fourth problem type analyzed in the thesis is the multi-mode resource constrained scheduling problem. MRCPSP aims to minimize the duration of manufacture or construction with limited resource in which activities of the project can be executed in more than one method. MRCPSP is usually faced in manufacture not in construction, because the MRCPSP requires detailed planning endeavor in order to provide the required data required for the analysis. Moreover, solution of MRCPSP is significantly difficult than the optimization problems mention so far in this thesis. Because of this, very talented planning engineers and experts should be employed to handle MRCPSP. Construction projects differs from manufacturing projects that construction projects are unique and executed only once while manufacturing projects are executed significantly many times. Because of this, exhaustive optimization procedures are not implemented for the solution of scheduling of construction projects. On the other hand, developed meta-heuristic algorithms GA, GASA and HGASA are good candidates for the solution of optimization problem of MRCPSP in construction sector.

The last problem type analyzed is the resource constrained time cost trade-off problem. This problem aims to minimize the total project cost without overriding the resource constraints. This problem is the combination of MRCPSP and TCT type problems. Solution of RCTCTP aims an overall optimization in resource demand and project cost. The two objective functions are simultaneously evaluated in RCTCTP. Similar to MRCPSP, RCTCP also requires detailed data about project and talented optimization algorithm. For this reason, RCTCTP is not analyzed in construction projects in which the tender documents are prepared before the final design is available.

In order to provide necessary data for the RCTCTP, automated quantity take-off and cost estimation systems should be used. There should be a database and a decision support system for the determination of different execution modes of the alternatives. Recently, optimum or near-optimum solution of RCTCTP was far from reality. However, the developed meta-heuristic algorithms can successfully be used for the improvement of the schedule during construction. The developed meta-heuristic algorithms presented good results. PSO and GA could not improve their initial results notably. HGASA and GASA converged into better results. Convergence of GASA is better than HGASA. Consequently, GASA present better results than the other meta-heuristic algorithms in this problem so it is the most suitable algorithm for this problem.

Although superior results are obtained when compared with the previous analysis results obtained from the literature. The analysis results obtained in this thesis study can be further improved by experimental design. The method can be implemented on SRCPSP, MRCPSP and RCTCT in order to improve the model parameters. Experimental design requires several computations of test problems with different parameter combinations. As a result of this, the endeavor of experimental design would be enormous. On the other hand, the analysis on TCT problem has shown that significant improvements in the convergence capability of the meta-heuristic algorithms can be obtained.

Multi-core processor desktop and laptop computers are standard. In order to benefit from the multi-core technology, the population based algorithms can be parallelized and benefited from the multi-core processor technology. Significant improvements can be obtained by parallel processing and the computation duration can be decreased without requiring another computer.

The CPM scheduler can schedule four logical relationships with positive and negative lags. Besides the simple CPM logical relationships, certain restrictions can be added for certain activities such as latest start or finish date for a certain activity. Furthermore, logical network relationships which define latest possible start or finish time of an activity can be defined which differs from the simple CPM logical relationship defining earliest possible start or finish time.

Philosophy “Strike, while the iron is hot” is valid both in optimization and life.

REFERENCES

Ackley D. H. (1987), "An empirical study of bit vector function optimization". Genetic Algorithms and Simulated Annealing, Morgan Kauffmann, Los Altos CA: 170 – 204.

Afshar A., Kaveh A., and Shoghli O. R. (2007), "Multi-objective optimization of time-cost-quality using multi-colony ant algorithm", Asian Journal of civil engineering (Building and Housing), Vol. 8 (2), pp. 113 – 124.

Afshar A., Ziaraty A. K., Kaveh A. and Sharifi F., (2009), "Nondominated Archiving Multicolony Ant Algorithm in Time-Cost Trade-Off Optimization", Journal of Construction Engineering and Management, Vol. 135 (7), pp. 668 – 674.

Ahuja H. N., (1976), "Construction performance controls by networks", Wiley, New York.

Agarwal R., Tiwari M. K. and Mukherjee S. K., (2007), "Artificial immune system based approach for solving resource constraint project scheduling problem", International Journal of Manufacturing Technology, Vol. 34, pp. 584 – 593.

Alcaraz J., Maroto C. and Ruiz R. (2003), " Solving the multi-mode resource constrained project scheduling problem with genetic algorithms", Journal of the Operational Research Society, Vol. 54, pp. 614 – 626.

Alcaraz J., Marato C. and Ruiz R. (2004), "Improving the performance of genetic algorithms for the RCPS problem", Proceedings of the ninth International Workshop on Project Management and Scheduling, Nancy, pp. 40 – 43.

Alcaraz J. , Maroto C. (2001), "A robust genetic algorithm for resource allocation in project scheduling", Annals of Operations Research 102 p. 83 – 109.

Al-Jibouri S., (2002), "Effect of Resource Management Regimes on Project Schedule", International Journal of Project Management Vol. 20, pp. 271 – 277.

Allam S. I. G., (1988), "Multi-project scheduling: a new categorization for heuristic scheduling rules in construction scheduling problems", Construction Management and Economics, Vol. 6 (2), pp. 93 – 115.

Antill J., Woodhead R., (1982), "CPMs in construction practice", Wiley-Interscience, New York.

Baar T., Brucker P., Knust S. (1998), "Tabu-search algorithms and lower bounds for the resource-constrained project scheduling problem", in: Voss S., Martello S., Osman I., Rouicairol C., (Eds.), Meta-heuristics: Advances and Trends in Local

Search Paradigms for Optimization, Kluwer Academic Publishers, Dordrecht, pp. 1-8.

Babu A. J. G., and Suresh N. (1996), "Project Management With Time, Cost, and Quality Considerations". Journal of Operational Research, 88: 320 – 327.

Balas E., (1970), "Machine sequencing: Disjunctive graphs and degree-constrained subgraphs", Naval Research Logistics Quarterly, Vol. 17 (1), pp. 1 – 10.

Balas E., (1971), "Project scheduling with resource constraints", Application of Mathematical Programming, The English University Press, pp. 187 – 200.

Ballentine L. B., (1999), "An introduction to design of experiments a simplified approach", ASQ Quality Press Milwaukee, Winsconsin.

Barber T. J., and Boardman J. T. (1988), "Knowledge –Based Project Control Employing Heuristic Optimisation". IEE Proceedings, 135(8): 529 – 538.

Bartusch M., (1983), "Optimierung von Netzplanen mit Anordnungsbeziehungen bei kanppen Betriebsmitteln, PhD Thesis, Technical University of Aachen.

Berger P. D. and Maurer R. E., (2002), "Experimental Design with applications in Management, Engineering, and the Sciences", Duxbury Thomson Learning, United States.

Besten M. D., Stützle T. and Dorigo M., (2000), "An ant colony optimization application to the single machine total weighted tardiness problem", Proceeding of the Sixth International Conference on Parallel Problem Solving from Nature, Springer, Berlin, Germany.

Bidhandi H. M. (2006), "A New Approach Based on the Surrogating Method in the Project Time Compression Problems". Ann Oper Res, 143: 237 – 250.

Blum C. and Samples M. (2004), "An ant colony optimization algorithm for shop scheduling problems." J Math Modelling Algorithms, Vol. 3(3), pp. 285 – 308.

Blum C. (2005a), "Beam-ACO – Hybridizing ant colony optimization with beam search: An application to open shop scheduling." Computers & Operations Res. Vol. 32 (6), pp. 1565 – 1591.

Blum C. (2005b), "Ant colony optimization: Introduction and recent trends", Physics of life reviews 2, 353 – 373.

Bock D. B. and Patterson J. H., (1990), "A comparison of due date setting, resource assignment, and job pre-emption heuristics for the multiproject scheduling problem", Decision Sciences, Vol. 21(2), pp. 387 – 402.

Boctor F. F., (1996), "A new and efficient heuristic for scheduling projects with resource restrictions and multiple execution modes", European Journal of Operational Research, Vol. 90(2), pp. 349 – 361.

Bouleimen K. and Lecocq H., (1998), "A new efficient simulated annealing algorithm for the resource-constrained project scheduling problem", Technical Report, Service de Robotique et Automatisation, Universite de Liege.

Bouleimen K. and Lecocq H. (2003), "A new efficient simulated annealing algorithm for the resource-constrained project scheduling problem and its multiple modes version", European Journal of Operational Research 149 pp. 268 – 281.

Brand J. D., Meyer W. L. and Shaffer L. R., (1964), "The resource scheduling problem in construction" 7th edition, Civil Engineering Studies, Report 5, Department of Civil Engineering, University of Illinois, Urbana.

Brucker P. and Knust S., (2003), "Lower bounds for resource-constrained project scheduling problems", European journal of operational research, Vol: 149, pp. 302 – 313.

Brucker P. and Knust S., (2000), "A linear programming and constraint propagation-based lower bound for the RCPSP", European Journal of Operational Research, Vol: 127, pp. 355 – 362.

Brucker P., Drexl A., Möhring R., Neumann K. and Pesch E. (1999), "Resource-constrained project scheduling: Notation, classification, models, and methods", European Journal of Operational Research 112 (1999) 3-41.

Brucker P., Knust S., Schoo A. and Thiele O. (1998), "A Branch and Bound Algorithm for the Resource-Constrained Project Scheduling Problem", European Journal of Operational Research, Vol. 107, pp. 272 – 288.

Burgess A. R. and Killebrew J. B., (1962), "Variation in activity level on a cyclical arrow diagram", Journal of Industrial Engineering, Vol. 13, pp. 76 – 83.

Burleson R. C., (1997), "An analysis of multiskilled labor strategies in construction", the University of Texas at Austin, Doctoral Dissertation, May 1997.

Burns S. A., Liu L., and Feng C. W., (1996), "The LP/IP hybrid Method for Construction Time-Cost Trade-off Analysis". Construction Management and Economics, 14: 265 – 276.

Butcher W. S., (1967), "Dynamic programming for project cost-time curve", Journal of Construction Division ASCE, Vol. 93, pp. 59 – 73.

Chan W. T., Chua D. K. H. and Kannan G., (1996), "Construction resource scheduling with genetic algorithms", Journal of Construction Engineering and Management, Vol. 122 (2), pp. 125 – 132.

Chassiakos A. P. and Sakellaropoulos S. P., (2005), "Time-cost optimization of construction projects with generalized activity constraints", Journal of Construction Engineering and Management, Vol. 131(10), pp. 1115 – 1124.

Chen D., Lee C. Y. and Park C. H. (2005), "Hybrid Genetic Algorithm and Simulated Annealing (HGASA) in Global Function Optimization". Proceedings of the 17th IEEE International Conference ın Tools with Artificial Intelligence (ICTAI'05), 1082-3409/05.

Chiu Y.S. P. and Chiu S. W., (2005), "Incorporating expedited time and cost of the end product into the product structure diagram", International Journal of Machine Tools & Manufacture Vol: 45 pp. 987 – 991.

Chua D. K. H., Chan W. T. and Govidan K., (1997), "A time-cost trade-off model with resource consideration using genetic algorithm", Civil Engineering and Environmental Systems, Vol. 14(4), pp. 291 – 311.

Coelho J., Tavares L. (2003), "Comparative analysis of meta-heuristics for the resource constrained project scheduling problem", Technical report, Department of Civil Engineering, Instituto Superior Tecnico, Portugal.

Damak N., Jarbouri B., Siarry P. and Loukil T., (2009), "Differential evolution for solving multi-mode resorce-constrained project scheduling problems", Computers & Operations Research, Vol. 36, pp. 2653 – 2659.

Das A. and Chakrabarti (2005), "Quantum annealing and related optimization methods", lecture notes in physics 679 (Springer, Berlin Heidelberg 2005).

Davis E. W., (1969), "An exact algorithm for the multiple constrained-resource project scheduling problem", PhD Dissertation, Yale University.

Davis E. W. and Heidorn G. E., (1971), "An algorithm for optimal project scheduling under multiple resource constraints", Management Science, Vol. 17(12), pp. B803 – B816.

Davis E. W., (1973), "Project Scheduling under Resource Constraints – Historical Review and Categorization of Procedures", IIE Transactions, 5:4, pp. 297 – 313.

Davis, E. W. and Patterson J. H., (1975), "A comparison of Heuristic and optimum solutions in resource-constrained project scheduling", management science vol. 4.

De Jong K. A. (1975), "An analysis of the behaviour of a class of genetic adaptive systems". Doctoral dissertation, University of Michigan.

De P., Dunne E. J., Ghosh J. B., and Wells C. E. (1995), "The discrete time-cost trade-off problem revisited". European Journal of Operational Research, 81: 225 – 238.

Dean B. V., Denzler D. R. and Watkins J. J., (1992), "Multiproject staff scheduling with variable resource constraints", IEEE Transactions on engineering management, Vol. 39(1), pp. 59 – 72.

Debels D., De Reyck B., Leus R., Vanhoucke M. (2006), “A hybrid scatter search /Electromagnetism meta-heuristic for project scheduling”, European Journal of Operational Research 169 (2) pp. 638 – 653.

Demeulemeester E., Reych B. D. and Herroelen W. (2000), “The discrete time/resource trade-off problem in project networks: a branch-and-bound approach” IEE Transactions 32, 1059 – 1069.

Demeulemeester E. L., Herroelen W. S., and Elmaghraby S. E. (1996), “Optimal Procedures for the Discrete Time Cost trade off Problem in Project Networks”. European Journal of Operational Research, 88: 50 – 68.

Demeulemeester E. and Herroelen W., (1992), “A Branch-and-Bound Procedure for the Multiple Resource-Constrained Project Scheduling Problem”, Management Science, Vol. 38, No. 12, pp. 1803 – 1818, December.

Den Besten M. L., Stützle T. and Dorigo M. (2000), “Ant Colony optimization for the total weighted tardiness problem” Peoceedings of PPSN-VI, sixth international conference on parallel problem solving from nature. Lecture notes in Comput Sci, vol. 1917. Berlin: Springer pp. 611 – 620.

Dorigo M. and Blum C., (2005), “Ant colony optimization theory: A survey”, Theoretical computer science 344, 243 – 278.

Dorigo M. and Stützle T., (2004), “Ant colony optimization”, A Bradford Book, The MIT Press, Cambridge, Massachusetts, London, England.

Dorigo M., Maniezzo V. and Colorni A., (1996), “Ant System: Optimization by a colony of cooperating agents”, IEEE Transactions on Systems, Man, and Cybernetics-Part B, Vol. 26(1), pp. 29 – 41.

Drexl, A. and Gruenewald, J. (1993), “Nonpreemptive multi-mode resource-constrained project scheduling”, IIE Transactions 25, pp. 74 – 81.

Easa, S. M., (1989), “Resource levelling in construction by optimization”, Journal of Construction Engineering and Management, Vol. 115, pp. 302 – 316.

Elbeltagi E., Hegazy T., and Grierson D. (2005), “Comparison Among Five Evolutionary-Based Optimization Algorithms”. Advanced Engineering Informatics, 19: 43 – 53.

Elmaghraby S. E., (1967), “On the expected duration of PERT type networks”, Management Science, Vol. 13 (5), pp. 299 – 306.

Elmaghraby S. E. and Cole R. T., “On the control of production in Small job-shops”, Journal of Industrial Engineering, Vol. 14(4), pp. 186 – 196.

Elmaghraby S. E., (1977), “Activity networks: Project planning and control by network models”, John Wiley & Sons.

Eshelman L.J., Caruana R. A., Schaffer J. D. (1989), “Biases in the crossover landscape”. Proceedings of 3rd International Conference on Genetic Algorithms. Morgan aufmann, Los Altos, CA: 10 – 19.

Eshtehardian E., Afshar A., and Abbasnia R. (2008), “Time-Cost Optimization: Using GA and Fussy Sets Theory For Uncertainties in Cost”. Construction Management and Economics, 26: 679 – 691.

Fairly A. (1991), “Comparison of methods of choosing the crossover point in the genetic crossover operation”. Department of Computer science , University of Liverpool.

Feng C. W., Liu L., and Burns S. A. (1997), “Using Genetic Algorithms to Solve Construction Time-Cost Trade-off Problems”. Journal of Computing in Civil Engineering, 11(3): 184 – 189.

Fisher M. L., (1973), “Optimal solution of scheduling problems using lagrange multipliers: Part I”, Operations Research, Vol. 21(5), pp. 1114 – 1127.

Fondahl J. M., (1961), “A non-computer approach to the critical path method for the construction industry”, Technical Report, No. 9, Construction Institude, Department of Civil Engineering, Stanford University.

Gagne C., Price W. L. and Gravel M., (2002), “Comparing an ACO algorithm with other heuristics for the single machine scheduling problem with sequence – dependent setup times.” J Oper Res Soc vol. 53, pp. 895 – 906.

Gonçalves J. F., Mendes J.J.M. and Resende M. G. C., (2008), “A genetic algorithm for the resource constrained multi-project scheduling problem”, European Journal of Operational Research, Vol. 189, pp. 1171 – 1190.

Gorenstein S., (1972), “An algorithm for project (job) sequencing with resource constraints”, Operations Research, Vol. 20(4), pp. 835 – 850.

Goyal S. K. (1996), “A Simple Time-cost Tradeoff Algorithm”. Production Planning & Control, 7(1): 104 – 106.

Goyal S. K. (1975), “A note on a simple CPM time-cost tradeoff algorithm”, Management Science Vol. 21 (6), pp. 718 – 722.

Hadley G., (1964), “Nonlinear and dynamic programming”,Addison-Wesley series in management science and economics, Massachusetts.

Hannan E. L., (1978), “The application of goal programming techniques to the CPM problem”, Socio-economic Planning Sciences, Vol. 12(5), pp. 267 – 270.

Hansen P., Oğuz C., Mladenovic N. (2008), “Variable neighbourhood search for minimum cost berth allocation”, European journal of operational research 191, 636 – 649.

Hansen P. and Mladenovic N. (2001), "Variable neighbourhood search: Principles and applications", European Journal of operational research, 130, p. 449 – 467.

Harris R. B., (1978), "Precedence and Arrow Networking Techniques for Construction", Wiley, New York.

Harris R. B., (1990), "Packing Method for resource levelling (pack)", Journal of Construction Engineering and Management, Vol. 116, pp. 39 – 43.

Hartmann S., (1998), "A competitive genetic algorithm for resource-constrained project scheduling", Naval Research Logistics 45 pp. 733 – 750.

Hartmann S., (2001), "Project scheduling with multiple modes: A genetic algorithm", Annals of Operations Research, Vol. 102, pp. 111 – 135.

Hartmann S. and Drexl A., (1998), "Project scheduling with multiple modes: A comparison of exact algorithms", Networks Vol. 32, pp. 283 – 297.

Hartmann S., (2002), "A self-adapting genetic algorithm for project scheduling under resource constraints", Naval Research Logistics 49 pp. 433 – 448.

Hegazy T. (1999), "Optimization of construction time-cost trade-off analysis using genetic algorithms", Canadian Journal of Civil Engineering, 26, pp. 685 – 697.

Hegazy T. (2001), "Computer-based construction project management" 1st Edition, Prentice Hall.

Hegazy T. and Kassab M., (2003), "Resource Optimization Using Combined Simulation and Genetic Algorithms", Journal of Constructio Engineering and Management, Vol. 129, No. 6, pp. 698 – 705, November/December.

Hiyassat M. A. S. (2000), "Modification of minimum moment approach in resource levelling", Journal of Construction Engineering and Management, Vol. 126, No. 4.

Hong C. (2006), "A Distributed Hybrid Heuristics of Mean Field Annealing and Genetic Algorithm for Load Balancing Problem". Lecture Notes in Computer Science, 726 – 735.

Hwang S. F., and He R.S. (2006), "Improving Real-Parameter Genetic Algorithm with Simulated Annealing for Engineering Problems". Advances in Engineering Software 37: 406 – 418.

Jarboui B., Damak N., Siarry P., and Rebai A. (2008), "A combinatorial particle swarm optimization for solving multi-mode resource-constrained project scheduling problems", Applied Mathematics and Computation Vol. 195, pp. 299 – 308.

Johnson T. J. R., (1967), "An algorithm for the resource constrained project scheduling problem", MIT PhD. Dissertation.

Jozefowska J., Mika M., Rozycki R., Waligora G. and Weglarz J., (2001), “Simulated annealing for multi-mode resource-constrained project scheduling”, Annals of operations research, vol. 102, pp. 137 – 155.

Kapsalis A., Smith G. D., and Rayward-Smith V.J. (1993), “Solving the graphical steiner tree problem using genetic algorithms”. The Journal of the Operational Research Society 44: 397 – 406.

Kandil A. and El Rayes K., (2006), “Parallel genetic algorithms for optimizing resource etilization in large-scale construction projects”, Journal of Construction engineering and Management, Vol. 132 (5), pp. 491 – 498.

Kanit R., Gunduz M. and Ozkan O. (2009), “Investigating the effectiveness of certain priority rules on resource scheduling of housing estate projects”, Journal of Construction Engineering and Management, Vol. 135 (7), pp. 609 – 613.

Kelly J. E., (1961), “Critical path planning and scheduling: mathematical basis”, Operations Research, Vol: 9 (3), pp. 167 – 179.

Kennedy J. and Eberhart R. (1995), “Particle swarm optimization”, 0-7803-2768-3/95, IEEE.

Khang D. B., and Myint Y. M. (1999), “Time, Cost and Quality Trade-off in Project Management: A Case Study”. International Journal of Project Management, 17(4): 249 – 256.

Kim J. L. and Ellis R. D., (2008), “Permutation-based elitist genetic algorithm for optimization of large-sized resource-constrained project scheduling”, Journal of construction engineering and management, Vol, 134 (11), pp. 904 – 913.

Kirkpatrick S., Gellat C. D., and Vecchi M. P., (1983), “Optimization by simulated annealing”. Science, 220, 671 – 680.

Klein R., (2000), “Project scheduling with time-varying resource constraints”, International Journal of Production Research 38 (16) pp. 3937 – 3952.

Klein R. and Scholl A., (1999), “Computing lower bounds by destructive improvement: An application to resource-constrained project scheduling”, European Journal of Operational Research, Vol: 112, pp. 322 – 346.

Kuang Y.P. and Xiong Y., (2006), “Construction time-cost trade-off analysis using ant colony optimization algorithm”, International Conference on Management Science and Engineering, ICMSE ’06.

Kochetov Y., Stolyar A., (2003), “Evolutionary local search with variable neighbourhood for the resource constrained project scheduling problem”, Proceedings of the 3rd International Workshop of Computer Science and Information Technologies, Russia.

Kolisch R., (1995), "Project scheduling under resource constraints – efficient heuristics for several problem classes", Physica 14.

Kolisch R., (1996a), "Efficient priority rules for the resource-constrained project scheduling problem", Journal of Operations Management 14 pp. (179 – 192).

Kolisch R., (1996b), "Serial and parallel resource-constraint project scheduling methods revisited: Theory and computation", European Journal of Operational Research 90 pp. 320 – 333.

Kolisch R., Drexl A., (1996), "Adaptive search for solving hard project scheduling problems", Naval Research Logistics 43, pp. 23 – 40.

Kolish R. and Sprecher A., (1996), "PSPLIB – A project scheduling problem library", European Journal of Operational Research, Vol. (96), pp. 205 – 216.

Kolish R. and Drexl A., (1997), "Local search for nonpreemptive multi-mode resource-constrained project scheduling", IIE Transactions, Vol. 29, pp. 987 – 999.

Kurtulus I. S. and Narula S. C., (1985), "Multi-project Scheduling: Analysis of Project Performance", IIE Transactions, Vol. 17(1), pp. 58 – 66.

Lee S. M., (1976), "Linear Optimization for Management", Petrocelli/Charter, New York.

Lee S. M., Olson D. L., (1984), "A zero-one goal programming approach to multi-project scheduling", Project Management Journal, Vol. 15, pp. 61 – 74.

Leon V. J., Ramamoorthy B., (1995), "Strength and adaptability of problem-space based neighborhoods for resource-constrained scheduling", OR Spektrum 17, pp. 173 – 182.

Leu S. S. and Yang C. H., (1999), "GA-Based Multicriteria Optimal Model for Construction Scheduling", Journal of Construction Engineering and Management, Vol. 125, No. 6, pp. 420 – 427, November/December.

Leu S. S., Yang C.H. and Huang J. C., (2000), "Resource levelling in construction by genetic algorithm-based optimization and its decision support system application", Automation in Construction 10, 27 – 41.

Lewis-Beck M. S., (1993), "Experimental Design & Methods", International Handbooks of Quantitative Applications in the Social Sciences, Volume 3, Sage Publications, London.

Li H., and Love P., (1997), "Using Improved Genetic Algorithms to Facilitate Time-Cost Optimization". Journal of Construction Engineering and Management, 123(3): 233 – 237.

Li H., Cao J. N., and Love P. E. D., (1999), “Using Machine Learning and GA to Solve Time-Cost Trade-off Problems”. Journal of Construction Engineering and Management, 125(5): 347 – 353.

Lova A., Tormos P., Cervantes M. and Barber F., (2009), “An efficient hybrid genetic algorithm for scheduling projects with resource constraints and multiple execution modes”, International Journal of Production Economics, Vol. 117 (2), pp. 302 – 316.

Lu M. and Lam H. C., (2008), “Critical path scheduling under resource calendar constraints”, Journal of Construction Engineering and Management, Vol: 134 (1), pp. 25 – 31.

Maniezzo V. and Mingozzi A., (1999), “The project scheduling problem with irregular starting time costs”, Operations Research Leters, Vol. 25(4), pp. 175 – 182.

Marti R., Laguna M. and Glover F., (2006), “Principles of scatter search”, European journal of operational research 169 p.359 – 372.

Martinez J. and Ioannou P., (1993), “Resource levelling based on the modified minimum moment heuristic”, Proceeding of the 5th International Conference on Computing in Civil and Building Engineering, ASCE, Reston, Va., pp. 287 – 294.

Mattila K. G. and Abrahan D. M., (1998), “Resource levelling of linear schedules using integer linear programming”. Journal of construction engineering and management, Vol. 124, No. 3, May/June, 1998.

Mendes J. J. M., Gonçalves J. F., Resende M. G. C., (2009), “A random key based genetic algorithm for the resource constrained project scheduling problem”, computers and operations research 36 pp. 92 – 109.

Merkle D., Middendorf M., Schmeck H., (2002), “Ant colony optimization for resource-constrained project scheduling”, IEEE Transactions on Evolutionary Computation 6 pp. 333 – 346.

Merz P. and Freisleben B., (1997), “A genetic local search approach to the quadratic assignment problem”, Proceedings of the seventh international conference on genetic algorithm (ICGA’ 97), pp. 465 – 472.

Meyer W. L. and Shaffer L. R., (1965), “Extending CPM fpr multiform project time-cost curves”, Journal of the construction division, ASCE, Proc. Paper 4319., Vol. 91, pp. 46 – 67.

Mingozzi A., Maniezzo V., Riciardelli S. and Bianco L., (1998), “An exact algorithm for the resource-constrained project scheduling problem based on a new mathematical formulation”, Management Science, Vol: 44 (5), pp. 714 – 729.

Mladenovic N. and Hansen P., (1997), “Variable Neighborhood search”, Computers Ops. Res. Vol. 24 (11), pp. 1097 – 1100.

Moder J. J., Phillips C. R. and Davis E. W., (1983), “Project Management with CPM, PERT and precedence diagramming”, Third edition, Van Nostrand Reinhold Company, England.

Mohanty R. P., Siddiq., (1989), “Multiple projects – multiple resources constrained scheduling: A multiobjective analysis”, Engineering Costs and Production Economics, Vol. 18(1), pp. 83 – 92.

Moodie C. L. and Mandeville D. E., (1966), “Project resource balancing by assembly line balancing techniques”, Journal of Industrial Engineering, Vol. 17, pp. 377 – 383.

Mori M., and Tseng C. C., (1997), “A genetic algorithm for multi-mode resource constrained project scheduling problem”, European Journal of Operational Research Vol. 100, pp. 134 – 141.

Moselhi O. and El-Rayes K., (1993), “Scheduling of repetitive projects with cost optimization”, Journal of Construction Engineering and Management, Vol. 119 (4), pp. 681 – 697.

Moselhi O. and Lorterapong P., (1993), “Least impact algorithm for resource allocation”, Canadian journal of civil engineering, Vol: 20(2), pp. 180 – 188.

Moussourakis J., and Haksever C., (2004), “Flexible Model for Time/Cost Tradeoff Problem”. Journal of Construction Engineering and Management, 130(3): 307 – 314.

Möhring R. H., Schulz A. S., Stork F. and Uetz M., (1999), “Resource-Constrained Project Scheduling: Computing Lower Bounds by Solving Minimum Cut Problems”, LNCS 1643, pp. 139 – 150.

Neumann K. and Zimmermann J., (2000), “Procedures for resource levelling and net present value problems in project scheduling with general temporal and resource constraints”. European Journal of Operational Research 127, 425 – 443.

Ng S. T. and Zhang Y., (2008), “Optimizing construction time and cost using ant colony optimization approach”, Journal of Construction Engineering and Management, Vol. 134 (9), pp. 721 – 728.

Nonobe K.and Ibaraki T., (2001), “Formulation and tabu search algorithm for the resource constrained project scheduling problem (RCPSP)”, Technical Report, Kyoto University.

Nonobe K., Ibaraki T., (2002), “Formulation and tabu search algorithm for the resource constrained project scheduling problem”, C. C. Ribeiro, P. Hansen (Eds.), Essays and Surveys in Metaheuristics, Kluwer Academic Publishers, pp. 557 – 588.

Norbis M. I. and Smith J. M., (1988), “A multiobjective, multi-level heuristic for dynamic resource constrained scheduling problems”, European Journal of Operational Research, Vol. 33(1), pp. 30 – 41.

Nudtasomboon N. and Randhawa S. U., (1997), “Resource-constrained project scheduling with renewable and non-renewable resources and time-resource tradeoffs”, Computers ind. Engng Vol. 32, No. 1, pp. 227 – 242.

Özdamar L., (1999), “A genetic algorithm approach to a general category project scheduling problem”, IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews, Vol: 29, pp. 44 – 59.

Padman R., Smith-Daniels D. E. and Smith-Daniels V. L., (1997), “Heuristic scheduling of resource-constrained projects with cash flows”, John Wiley & Sons, Inc. Naval Research Logistics 44: 365 – 381, 1997.

Padilla E. M., Carr R. I., (1991), “Resource strategies for dynamic project management”, Journal of Construction Engineering and Management, Vol. 117(2), pp. 279 – 293.

Panagiotakopoulos, D., (1977), “A CPM time-cost computational algorithm for arbitrary activity cost functions”, INFOR 15, 183 – 195.

Patterson J. H., (1973), “Alternate methods of project scheduling with limited resources”, Naval Research Logistics Quarterly, Vol. 20 (4), pp. 767 – 784.

Patterson J. H. and Huber W. D., (1974), “A Horizon-varying, zero-one approach to project scheduling” Management Science, Vol. 20(6), pp. 990 – 998.

Patterson J. H. and Roth G. W., (1976), “Scheduling a project under multiple resource constraints: A zero-one programming approach”, IIE transactions, Vol. 8(4), pp. 449 – 455.

Patterson J. H. (1984), “A comparison of exact approaches for solving the multiple constrained resource, project scheduling problem”, Management Science, Vol. 30, No. 7, July 1984.

Patterson J. H., Slowinski R., Talbot F. B. and Weglarz J., (1989), “An algorithm for a general class of precedence and resource constrained scheduling problems”, Advances in Project Scheduling, Elsevier, Amsterdam.

Peteghem V. V. and Vanhoucke M., (2009), “A genetic algorithm for the preemptive and non-preemptive multi-mode resource-constrained project scheduling problem”, European Journal of Operational Research, Article in Press, Corrected Proof – Note to users.

Pourghaderi A. R., (2008), “Scatter search for multi-mode resource-constrained project scheduling problems”, IEEE International Conference on Industrial Engineering and Engineering Management, 8 – 11 December, Singapore.

Pritsker A. B., Watters L. J. and Wolfe P. M., (1969), "Multiproject scheduling with limited resources: A zero-one programming approach", Management Science, Vol.16, pp. 93 – 108.

Ranjbar M., De Reyck B., Kianfar F., (2009), "A hybrid scatter search for the discrete time/resource trade-off problem in project scheduling", European Journal of Operational Research Vol. 193 (1), pp. 35 – 48.

Ranjbar M., (2008), "Solving the resource-constraint project scheduling problem using filter-and-fan approach", Applied mathematics and computation, Vol. 201, pp. 313 – 318.

Reeves C. R., (1992), "A genetic algorithm approach to stochastic flowshop sequencing", IEE, London, Vol. 106, pp. 131 – 134.

Reeves C. R., (1995a), "Advanced topics in computer science modern heuristic techniques for combinatorial problems". McGraw-Hill Book Company Europe.

Reeves C. R., (1995b), "A genetic algorithm for flowshop sequencing". Computers & Operational Research.

Ryan T. P., (2007), "Modern Experimental Design", John Wiley & Sons, New Jersey.

Savin D., Alkass S. and Fazio P., (1997), "A procedure for calculating the weight-matrix of a neuronal network for resource levelling", Advances in Engineering Software, Vol. 28 pp. 277 – 283.

Savin D., Alkass S. and Fazio P., (1998), "Calculating Weight Matrix of Neural Network for Resource Leveling", Journal of Computing in Civil Engineering, Vol. 12(4), pp. 241 – 248.

Schirmer A., (2000), "Case-based reasoning and improved adaptive search for project scheduling", Naval Research Logistics, 47 pp. 201 – 222.

Schrage L., (1970), "Solving resource-constrained network problems by implicit enumeration-nonpreemptive case", Operations Research, Vol. 18(2), pp. 263 – 278.

Schwarze, J., (1980), "An algorithm for hierarchial reduction and decomposition of a direct graph", Computing 25, 47 – 57.

Seibert J. E. and Gerald W. E., (1991), "Time-Constrained Resource Leveling", Journal of Construction Engineering and Management, Vol. 117, No. 3, pp. 503 – 520.

Sepil C. and Ortac N., (1997), "Performance of the heuristic procedures for constrained projects with progress payments", Journal of the Operational Research Society 48, 1123 – 1130.

Senouci A. B., and Adeli H., (2001), “Resource scheduling using neural dynamics model of Adeli and Park”, Journal of Construction engineering and management, Vol. 127, No.1,pp. 28 – 34.

Shaffer L. R., Ritter J. B. and Meyer W. L., (1965), “The critical-path method”, McGraw-Hill Inc., U. S.

Shelokar P.S., Siarry P., Jayaraman V. K. and Kulkarni B. D., (2007), “Particle swarm and ant colony algorithms hybridized for improved continuous optimization”, Applied mathematics and computation 188, 129 – 142.

Shi Y. and Eberhart R., (1998), “A modified particle swarm optimizer”,0-7803-4869-9/98, IEEE.

Siemens N., (1971), “A Simple CPM Time-Cost Trade off Algorithm”. Management Science, 17(6): B-354 – B-363.

Slowinski R., (1981), “Multiobjective network scheduling with efficient use of renewable and non-renewable resources”, European Journal of Operational Research, Vol. 7(3), pp. 265 – 273.

Slowinski R., Soniewicki B. and Weglarz J., (1994), “DSS for multiobjective project scheduling”, European Journal of Operational Research, Vol. 79(2), pp. 220 – 229.

Sprecher A., Hartmann S. and Drexl A., (1997), “An exact algorithm for the project scheduling with multiple modes”, OR Spektrum, Vol. 19, pp. 195 – 203.

Sprecher A. and Drexl A., (1998), “Solving multi-mode resource-constrained project scheduling problems by a simple, general and powerful sequencing algorithm”, European Journal of Operational Research”, Vol. 107, pp. 431 – 450.

Speranza M. G. and Vercellis C., (1993), “Hierarcial models for multi-project planning and scheduling”, European Journal of Operational Research, Vol. 64(2), pp. 312 – 325.

Son J. and Skibniewski M. J., (1999), “Multiheuristic approach for resource levelling problem in construction engineering: Hybrid approach”. Journal of Construction engineering and management, Vol. 125, No. 1, January/February, 1999.

Stevens J. D., (1990), “Techniques for construction network scheduling”, McGraw-Hill, New York.

Stinson J. P., (1976), “A branch and bound algorithm for a general class of multiple resource-constrained scheduling problems”, PhD thesis, Graduate School of Business Administration, University of North Carolina.

Stinson J. P., Davis E. W. and Khumawala B., (1978), “Multiple resource-constrained scheduling using branch and bound”, AIIE Transactions, Vol. 9.

Stützle T., (1998), “An ant approach to the flow shop problem.” Proceedings of the 6th European congress on intelligent techniques & soft computing (EUFIT’ 98), Aachen: Verlag Mainz, pp. 1560 – 1564.

Suhanic G., (2001), “Computer-Aided Project Management”, first addition, Oxford University Press, Inc. New York.

Sun R., Tatsumi S. and Zhao G., (2001), “Multiagent reinforcement learning method with an improved antcolony system”, IEEE International Conference on Systems, Man, and Cybernetics, Vol. 3, pp. 1612 – 1617.

Talbot F. B., (1976), “An integer programming algorithm for the resource-constrained project scheduling problem”, PhD. Dissertation, Pennsylvania Satate University.

Talbot F. B. and Patterson J. H., (1978), “An efficient integer programming algorithm with network cuts for solving resource-constrained scheduling problems”, Management Science, Vol. 24, pp. 1163 – 1174.

Talbot F. B., (1982), “Resource-constrained project scheduling with time-resource tradeoffs: the non-preemptive case”, Management Science, Vol. 28, pp. 1197 – 1210.

Tareghian, H. R., and Taheri S. H., (2007), “A solution procedure for the discrete time, cost and quality trade off problem using electromagnetic scatter search”. Applied Mathematics and Computation, 190: 1136 – 1145.

Tchao C. and Martins S. L., (2008), “Hybrid Heuristics for multi-mode resource-constrained project scheduling”, Lecture Notes in Computer Science, Springer, Vol. 5313, pp. 234 – 242.

Toklu Y. C., (2002), “Application of Genetic Algorithms to Construction Scheduling With or Without Resource Constraints”, Can. J. Civ. Eng., Vol. 29, pp. 421 – 429.

Tormos P., Lova A., (2001), “A competitive heuristic solution technique for resource-constrained project scheduling”, Annals of Operations Research 102 pp. 65 – 81.

Tormos P., Lova A., (2003a), “An efficient multi-pass heuristic for project scheduling with constrained resources”, International Journal of Production Research 41 (5) pp. 1071 – 1086.

Tormos P., Lova A., (2003b), “Integrating heuristics for resource constrained project scheduling: One step forward.” Technical report, Departent of Statistics and Operations Research, Universidad Politecnica de Valencia.

Tseng L. Y. and Chen S. C., (2006), “A hybrid metaheuristic for the resource-constrained project scheduling problem”, European journal of Operational Research, Vol. 175, pp. 707 – 721.

Valls V., Ballestin F., Quintanilla M. S., (2005), “Justification and RCPSP: A technique that pays”, European Journal of Operational Research”, 165 pp. 375 – 386.

Valls V., Ballestin F., Quintanilla M. S., (2003), “A hybrid genetic algorithm for the RCPSP.” Technical report, Department of Statistics and Operations Research, University of Valencia.

Vanhoucke M., and Debels D., (2007), “The Discrete Time/Cost Trade off Problem: Extensions and Heuristic Procedures”. J Sched (2007) 10: 311 – 326.

Vanhoucke M., (2005), “New Computational Results for the Discrete Time/Cost Trade-off Problem With Time-Switch Constraints”. European Journal of Operational Research, 165: 359 – 374.

Wang L. and Zheng D.Z., (2001), “An effective hybrid optimization strategy for job-shop scheduling problems”, Computers & Operations Research Vol. 28, pp. 585 – 596.

Wei C. C., and Wang C. M. F., (2003), “Efficient Approach of Linearization in Project Compression”. Computers & Industrial Engineering, 44: 695 – 706.

Wei C. C., Liu P. H. and Tsai Y. C., (2002), “Resource-constrained project management using enhanced theory of constrained”, International Journal of Project Management, Vol. 20, pp. 561 – 567.

Weist J. D., (1967), “Heuristic model for scheduling large projects with limited resources”, management science, vol. 2.

Weist J. and Levy F., (1977), “A Management guide to PERT/CPM with GERT / PDM / DCPM and other networks”, Englewood Cliffs, Prentice Hall.

Woodworth B. M. and Willie C. J., (1975), “A heuristic algorithm for resource levelling in multi-project, multi-resource scheduling”, Decision Sciences Vol. 6, 525 – 540.

Xiong Y. and Kuang Y., (2008), “Applying an ant colony optimization algorithm-based multiobjective approach for time-cost trade-off”, Journal of Construction engineering and management, Vol. 134 (2), pp. 153 – 156.

Ya-ping K., and Ying X., (2006), “Construction Time-Cost Trade-off Analysis Using Ant Colony Optimization Algorithm”. International Conference on Management Science and Engineering, 2006. ICMSE’06, 2039 – 2044.

Yang I. T., (2005a), “Impact of Budget Uncertainty on Project Time-Cost Tradeoff”. IEEE Transactions on Engineering Management, 52(2): 167 – 174.

Yang I. T., (2005b), “Chance-Constrained Time-Cost Tradeoff Analysis Considering Funding Variability”. Journal of Construction Engineering and Management, 131(9):1002 – 1012.

Zhang H., Tam C. and Li H., (2006), “Multimode project scheduling based on particle swarm optimization”, Computer-aided Civil and Infrastructure Engineering, Vol. 21, pp. 93 – 103.

Zheng, D. X. M., Ng, S. T., and Kumaraswamy, M. M., (2004), “Applying a Genetic Algorithm-Based Multiobjective Approach for Time-Cost Optimization”. Journal of Construction Engineering and Management, 130 (2): 168 – 176.

Zheng, D. X. M., Ng, S. T., and Kumaraswamy, M. M., (2005), “Applying Pareto Ranking and Niche Formation to Genetic Algorithm-Based Multiobjective Time-Cost Optimization”. Journal of Construction Engineering and Management, 131(1): 81 – 91.

Zhu G., Bard J. and Tu G., (2006), “A branch-and-cut procedure for the multimode resource-constrained project-scheduling problem”, Journal on Computing Vol. 18 (3), pp. 377 – 390.

CURRICULUM VITAE

Önder Halis BETTEMİR

Personal Information	
Nationality	Republic of Turkey
Address	645. Sokak Akşar Evleri No:54 06530 Konutkent Ankara Turkey
Home Phone	312 240 52 52
Mobile Phone	539 748 91 62
e-mail	onder.halisbettemir@gmail.com
TR Identity No	71326103058
Date of Birth	07.09.1979
Place of Birth	Bursa
Driving License	Type B (1999)
Education	PhD
Marital Status	Single
Working Experience	6 Years
Employment	Employed
Military State	Postponed (31.03.2012)

Educational Background

PhD 09.2006 – 07.2009	**Middle East Technical University** Faculty of Engineering, Civil Engineering	3,71 / 4
Master of Science 09.2003 – 09.2006	**Middle East Technical University** Faculty of Engineering, Civil Engineering	3,43 / 4
Undergraduate 09.1999 – 06.2003	**Middle East Technical University** Faculty of Engineering, Civil Engineering	2,89 / 4
High School 09.1994 – 06.1997	**Private Arı Science School**	

Working Experience

09. 2006 –	**METU Construction Management Division** Research Assistant	Ankara

I am working as a research assistant in Middle East Technical University at Civil Engineering department. I have sound theoretical knowledge about engineering economy, construction planning, construction site management, quality control, progress payment, quantity take off and bidding.

I am able to use Primavera and MS Project software efficiently.

I have sound knowledge on finding optimum or near-optimum solutions of time cost trade-off, resource leveling, resource constrained scheduling and resource constrained time cost trade-off analysis.

I have gained knowledge on quantity take-off and cost estimation of housing insulation.

01.2004 – 09.2006 **METU Geodesy & Photogrametry Division** Ankara
Research Assistant

I had worked as a research assistant in Middle East Technical University at Civil Engineering department until September 2006. I have detailed theoretical and practical knowledge about Geodesy, Photogrammetry, Geographic Information System, Remote Sensing and surveying.

During the commission as a research assistant at geodesy and Photogrammetry I had gained high level of skills on surveying. I am able to perform positioning by both hand held and geodetic Global Positioning System (GPS) receivers. In addition to this I can perform surveying by Total Station and Level and analyze the measurements.

Software I can use about Geodesy and Photogrammetry:
Bernese: Interpretation of GPS measurements and positioning

ArcGIS 9.2, MapInfo 7.0: Applications related with Civil Engineering such as, quantity take off (amount of cut&fill of earthwork, concreting, walls by using CAD drawings), network analysis.

Matlab: I am able to use Matab mathematical programming language effectively at Geodesy Photogrammetry, remote sensing, positioning via GPS, analyzing fieldwork and precise orbit determination.

During my study on Master of Science I had generated codes for the coordinate transformation from UTM to WGS84, local reference systems to inertial or earth centered earth fixed systems.

I had developed software by using Matlab which can perform rectification of images taken by airborne camera or satellite. I had developed codes on Matlab which can perform “false coloring” or image enhancements. In addition to this, I had generated software which can perform resampling by nearest neighborhood algorithm. I had developed codes which can read and compute the position the GPS absolute positioning measurements in RINEX format.

PCI Geomatica: Rectification of satellite images and image enhancement.

Besides these subjects I have sound knowledge on traverse and adjustment of observations.

PUBLICATIONS

1. Karslioglu M. O., Friedrich J., Bettemir Ö.H., (2006), “Orthorectification of Monoscopic BilSAT images by a new differential image rectification model”, ISPRS WG I/5 & I/6 Workshop on Topographic Mapping from space (with special emphasis on small satellites) February 14 – 16, Ankara Turkey.
2. Bettemir Ö. H., Karslioglu M. O., Friedrich J., (2007), “Error analysis and testing of DRM for frame cameras”, Recent Advances in Space Technologies 2007.
3. Bettemir Ö. H., (2008) “Differential sensitivity analysis for the accuracy estimation of orthorectification of small satellite images”, Proceedings of the international workshop on Small Satellites, New Missions and New Technologies SSW 2008, June 05 – 07, 2008, Istanbul Turkey.
4. Bettemir Ö. H., and Karslıoglu M. O., (2008), “BilSAT 1 uydu görüntüleri için diferansiyel rektifikasyon parametrelerinin düzeltilmesi”, UZAL – CBS 2008, 2. Uzaktan Algılama ve Coğrafi Bilgi Sistemleri Sempozyumu-2008, 13 – 15 Ekim 2008, Kayseri.
5. Bettemir Ö. H., and Sönmez, R. (2008), “Automated cost estimation of dam projects with geographic information systems”, 8th International Congress on Advances in Civil Engineering, 15 – 17 September 2008, Eastern Mediterranean University, Famagusta, North Cyprus, Volume 2, pp. 557 – 561.
6. Bettemir Ö. H., (2009), “Differential sensitivity analysis for the accuracy estimation of orthorectification of small satellite images”, Proceedings of 4th international conference on Recent Advances in Space Technologies RAST 2009, June 11 – 13, 2009 Istanbul Turkey.
7. Bettemir Ö. H., Sonmez R., (2009), “Estimation of building costs using geographic information systems”, fifth international conference on construction in the 21st century (CITC-V) “Collaboration and Integration in Engineering, Management and Technology”, May 20 – 22, 2009, Istanbul, Turkey.