UNIVERSITY OF BAHRAIN

College of Engineering

Department of Mechanical Engineering



Optimum Planning and Scheduling of Repetitive Construction Projects Using Multi-Objective Cuckoo Search Algorithm

A Thesis Submitted in Partial Fulfilment of the Requirements for the Master Degree in Engineering Management

Submitted by

Saraa Naseer Kadhim Alasadi

University Number: 20074441

Supervised by

M. P. Saka

(Professor)

University of Bahrain

Kingdom of Bahrain

September 2015

University of Bahrain
Graduate Studies Council

Thesis Discussion Committee Decision

Kadhim Alasadi

Student Name: Saraa Nasear A	Student ID: 20074441
Department: Mechanical Eng.	College: Engineering
Program: Engineering Management Master Program	
Thesis Title: Optimum Planning and Scheduling of Repetitive Construction Projects Using Multi-Objective Cuckoo Search Algorithm.	
Supervisor Name: Prof. M. P. Saka	Student GPA: 3.77

The Discussion Committee met on 08/09/2015 (day and date) _____ in Hall no. 14-162 (hall/room no.) _____. Upon discussing the student whose details are stated above in public, the discussion committee recommends the following:

The student awarded grade B+ In the thesis subject Engineering Management

☐ Without making any changes to the thesis
☐ After making the modifications attached with this decision

Discussion Committee Members Approval:

Name	Signature
1. Prof. M.P. Saka (Advisor)	
2. Prof. Saad Suliman (Internal Examiner)	
3. Prof. Murat GUNDUZ (External Examiner)	

قسم الهندسة الميكانيكية
DEPT. OF MECHANICAL ENGINEERING
UNIVERSITY OF BAHRAIN

Abstract

Most of the construction projects such as infrastructure systems and housing projects are repetitive in nature. Therefore it becomes very important to schedule their work in such a way that continuity of their work will not be disturbed with interruptions. In scheduling construction projects, time and cost are two important coupled factors that affect the whole project. Shortening project duration results in increase in the project cost. It is apparent that an optimum balance is required to be determined between these two project objectives. In this study multi-objective optimization model for resource utilization is developed for planning and scheduling of repetitive construction projects. The model minimizes concurrently the project duration and project cost. The model obtains set of Pareto-optimal solutions that represent the trade-off between project duration and project cost. Project manager will be able to select one of the solutions depending on a quantitative assessment of the trade-off between time and cost that satisfies the requirements of the project under consideration. Cuckoo search algorithm which one of the recent and efficient meta-heuristic optimization technique inspired by the behaviour of self-organized systems. Specifically, the cuckoo search (CS) is a numerical optimization algorithm was inspired by the obligate brood parasitism of some cuckoo species when they lay their eggs in the nests of birds from other species -host birds-.

A Case study will be conducted based on a housing project in Bahrain to demonstrate and validate the algorithm then the results and findings are reported.

Dedication

I dedicate my thesis to all my family and my husband who have never left my side. A special feeling of gratefulness to my devoted parents, Naseer and Yousra whose words of encouragement for tenacity rings in my ears since my childhood.

Acknowledgment

I thank all my family members for their constant support and patience during my studies in all my life. My gratitude and appreciation of this work supervision is for Prof. M. P. Saka for his continuous guidance and generous assistance. I extend this gratitude to Prof. Saad Suliman for his upkeep guidance in the Masters' Degree in Engineering Management.

List of Contents

Subject	Page
Thesis Discussion Committee Decision	**i**
Abstract	**ii**
Dedication	**iii**
Acknowledgment	**iv**
List of Contents	**v**
List of Tables	**viii**
List of Figures	**x**
List of Appendices	**xiii**
Chapter 1 **Introduction to the Study**	
1.1 Introduction	**2**
1.2 Research Problem	**3**
1.3 Research Questions	**4**
1.4 Research Objectives	**4**
1.5 Importance of the Research	**5**
1.6 Limitations of the Research	**6**
1.7 Terminologies of the Research	**7**
1.7.1 Abbreviations of the Research	**7**
1.7.2 Symbols of the Research	**8**
Chapter 2 **Theoretical Framework and Previous Studies**	
2.1 Optimization	**12**
2.1.1 Optimization Classifications	**13**
2.1.2 Meta-heuristic Algorithms	**16**
2.2 Meta-Heuristics in Construction Management	**35**
2.2.1 Cost Estimation in Construction Projects	**35**
2.2.2 Project Scheduling	**36**
2.2.3Time-Cost Trade-Off with Meta-Heuristics	**36**

2.2.3 Previous Studies in Multi-Objective Cost-Time Trade-	**41**
Chapter 3 **Methods and Procedures**	
3.1 Problem Formulation and Modelling	**49**
3.2 Solution Approach (MOCS)	**50**
3.3 Experimentations and Results	**51**
3.4 Discussion and Conclusion	**52**
Chapter 4	
Problem Formulation and Modelling	
4.1 Parameters	**54**
4.2 Decision Variables	**56**
4.3 Scheduling Module	**57**
4.4 Direct Cost Module	**63**
4.4.1 Material Cost	**63**
4.4.2 Labour Cost	**64**
4.4.3 Equipment Cost	**65**
4.4.4 Interruption Cost	**66**
4.5 Modelling of Repetitive Construction Project Schedule	**66**
Chapter 5 **Solution Approach (MOCS)**	
5.1 Optimization	**71**
5.1.1 Multi-objective Optimization (MOOP)	**71**
5.2 Optimization Using Cuckoo Search Algorithm	**79**
5.2.1 Cuckoo Birds Lifestyle	**80**
5.2.2 Single Objective Cuckoo Search Algorithm (CSA)	**83**
5.2.3 Multi-Objective Cuckoo Search Algorithm (MOCS)	**91**
Chapter 6 **Experimentations and Results**	
6.1 Computational Applications	**109**
6.1.1 Time-Cost Trade-off Case studies	**109**

Chapter 7 **Discussion and Conclusions**	
7.1 Conclusion	**144**
7.2 Recommendations	**147**
7.3 Future Work	**148**
References	**149**
Appendices	**153**
Biography	**152**

List of Tables

Number	Subject	Page
1	Quantity of materials in cubic meters	55
2	Productivities of each crew formation in cubic meters per day	55
3	Labour cost of each crew formation in USD per day	55
4	Equipment cost of each crew formation in USD per day	56
5	Solutions obtained for three span concrete bridge by (Hyari et. al, 2009)	117
6	Solutions obtained by the MOCS	118
7	Total costs of solutions obtained by (Hyari et. al, 2009)	123
8	Total costs of solutions obtained by MOCS	124
9	Quantity of materials in cubic meters per each unit	128
10	Productivities of each crew formation in cubic meters per day	128
11	Labour cost of each crew formation in BHD per day	129
12	Equipment Cost of each crew formation in BHD per day	129
13	Pareto solutions of Case Study 2A for a sample of 5 villas time and cost Trade-off	131
14	Pareto solutions for Case Study 2A for whole units' time-cost trade-off	135

15	Pareto solutions of Case Study 2B for a sample of 5 villas	140
16	Pareto Solutions for Case study 2B for 156 units' time-cost trade-off	142

List of Figures

Number	Subject	Page
1	Classifications of optimization (Yang, 2010)	13
2	Classification of algorithms	17
3	Classifications of Meta-heuristics (Dréo, 2011)	18
4	Meta-heuristics used to optimize construction management objectives (Liao et. al, 2011)	20
5	Timeline of meta-heuristics (Gandomi et. al, 2011)	22
6	One generation of evolutionary algorithm (Talbi, 2009)	23
7	Evolutionary algorithm template (Talbi, 2009)	24
8	Normal crossover (Talbi, 2009)	27
9	Meta-heuristics used in solving construction projects problems in literature (Liao et. al, (2011)	39
10	Quantities of work, available crew formation options and their related costs and productivities (Hyari et. al, 2009)	40
11	Scheduling model calculation for foundation activity (Hyari and El-Rayes, 2006)	44
12	Hyari's model development (Hyari et. al, 2009)	45
13	Comparison between three Pareto solutions (Afshar et. al, 2009)	46
14	Comparison of Pareto curves obtained by different models	46
15	Pareto solutions	72
16	Illustration of the decision variable space and	74

	the corresponding objective space	
17	Steps needed to find one final solution in MOOP	74
18	Population of 6 solutions and its non-dominated set (Deb, 2004)	75
19	Two goals in MOOP	77
20	Distribution and Conversion (Deb, 2001)	77
21	Pareto-optimal for four arrangements of two types of objectives (Deb, 2001)	78
22	Illustration of local such as solution (B) and global Pareto-optimal solutions (Deb, 2001)	79
23	Cuckoo Bird from (BBC Wildlife Video) (Yang, 2010)	80
24	Single Objective Cuckoo Search (CS) Pseudo-code (Yang and Deb, 2010)	86
25	Flow chart of CSA used by (Rajabioun, 2011)	87
26	2D Lévy flights in 50 steps (Yang, 2010)	89
27	2D Lévy flights in 100 steps (Yang and Deb, 2011)	90
28	Distribution of 100 consecutive steps (Yang and Deb, 2011)	90
29	MOCS pseudo-code (Yang and Deb, 2011)	92
30	MOCS Flow Chart	95
31	Pareto Front of the illustrating example	98
32	Pareto fronts of test functions SCH (Yang and Deb, 2011)	101
33	Obtained Pareto fronts of test functions SCH in this study	101
34	Pareto fronts of test functions ZDT1 (Yang and	103

	Deb, 2011)	
35	Obtained Pareto fronts of test functions ZDT1 in this study	103
36	Pareto fronts of test functions ZDT2 (Yang and Deb, 2011)	105
37	Obtained Pareto fronts of test functions ZDT2 in this study	105
38	Obtained Pareto fronts of test functions ZDT3	107
39	The bridge case study (Liu and Wang , (2007)	113
40	Pareto front of MOCS and the one obtained by (Hyari et. al, 2009)	116
41	Comparison between rounded results obtained by MOCS and by (Hyari et. al, 2009)	121
42	Pareto front for Case Study 2A of 5 units time-cost trade-off using MOCS and GA	130
43	Pareto solutions Case Study 2A of 156 units' time-cost trade-off	136
44	Comparison between obtained Pareto fronts with the contractor choice	137
45	Pareto optimum for Case Study 2B for 5 villas	139
46	Pareto front for Case study2B of whole units' time-cost trade-off	141

List of Appendices

Number	Appendix
1	MOCS Algorithm Coded Using MATLAB to solve Testing Problems
2	MOCS Algorithm Coded Using MATLAB to Solve Case Studies
3	Data of Three Contractors (Class A)
4	Time and Cost Modules
5	Parameters Used in Case Studies:
6	Darkulaib Housing Project Outline
7	Contractor Data for Darkulaib Housing Project

Chapter 1

Introduction to the Study

1.1 Introduction

1.2 Research Problem

1.3 Research Questions

1.4 Research Objectives

1.5 Importance of the Research

1.6 Limitations of the Research

1.7 Terminologies of the Research

Chapter 1

Introduction to the Study

In this chapter, an introduction is made for the research topic to briefly justify the need for studying the research problem.

1.1 Introduction

Repetitive construction projects with "repetitions in some or all of the performed construction activities in different locations or units in the project" are characterizing all projects. For Example, precast slabs crew needs to move from one unit to another to cover this activity in the entire project. Repetitive construction projects are clear in (1) infrastructure projects such as "highway projects, tunnels and bridges, railways, pipeline networks, and electrical and communication cables and in (2) building projects (e.g. high-rise buildings, apartments and housing development projects)". The repetitive projects are unique for crew work continuity which is not the case in "traditional network scheduling methods such as critical path method for scheduling construction projects" (Hyari et. al, 2009). Crew or resources continuity means "uninterrupted and unchanged" or keeping continuity and consistency in the usage of resources moving from a unit to another (Zhang et. al, 2013). Its advantages for project managers are best benefits from the "learning curve effect for each crew" and minimizing of the "off-on movement of crews" throughout the units project. Moreover, work interruption leads to an increased direct cost because of the idle crew time. Repetitive projects are called Non-serial only when contain subprojects or mini-projects of

similar activities, but with independent starting and finishing dates (Ezeldin and Soliman , 2009).

1.2 Research Problem

Generally, project management begins with the start of a project and continues throughout the project life cycle to truly optimize time and cost objectives. Typically, mentioned a project life cycle phases to be (1) initialization of an idea, followed by (2) engineering initial and detailed design drawings, (3) planning to choose crew formations, subcontracting, as well as, resource constraint planning, and project assessment methods (4) implementation phase including, scheduling, budgeting, and resource allocation, (5) monitoring by involving corrective actions, and finally (6) termination and evaluation.

Construction project management aims basically to satisfy and trade-off between three conflicting objectives which are time (schedule), cost, and performance (quality). One of the "most important aspects of project management" is the time and cost trade-off i.e. the less costs endured in resources allocated to an activity, more time will be required to complete it. (Liao et. al, 2011)

The reseach problem minimizes the project time and direct cost as two objective functions by finding the best crew formation and interruptions. A recent algorithm called Multi-Objective Cuckoo Search (MOCS) Algorithm is used in this optimisation problem to determine its solution due to its proven efficiency. The MOCS finds set of feasible solutions that are with minimum direct cost and minimum duration. Each of solution corresponds to certain crew formations and interruptions days.

1.3 Research Questions

There are several questions answered in this research:

- Does the Cuckoo Search Algorithm provide better trade-off solutions than that made by (Hyari et. al, 2009)? And does the Pareto optimum solution provide better solution that what made actually by the constructor?
- Is this study applicable in large projects such as a housing project of 156 villas?
- Does this trade-off support decision makers find the best contractor for the construction project? And who is the best contractor can construct the housing project bade on the time-cost trade-off.

1.4 Research Objectives

The objective of this research is to develop a proper multi-objective optimization model to plan and allocate resources of repetitive construction project problem to:

1. Provide trade-off solutions for repetitive activities in construction projects i.e. solutions that minimize the total construction cost and as well as-the total construction time.
    a. Best construction options or methods.
    b. Best interruption days in activities after repeating units in the construction project.

2. Apply the MOCS algorithm to optimise the trade-off in two projects i.e. three spans concrete bridge project and a housing project.
3. Use the model to trade-off by finding best contractors who can carry out the project. This will develop the usage of the trade-off model to support pre-selection of contractors for negotiation or tendering.
4. Provide a tool to roughly estimating cost and time of repetitive activities using the crew formations and interruption data.

1.5 Importance of the Research

This study is made due to the challenge of trade-off both time and cost or find solutions that satisfy both of these two important objectives. The research topic is in the field of construction project management that aims to find solutions that minimizes both cost and time in construction projects. The research aims to find better solutions for the benchmark problem –three span concrete bridges- than that made by (Hyari et. al, 2009). Furthermore the model used in this study will be applied in a real housing project in Bahrain.

The main contribution in this research is to use the MOCS Algorithms to solve time-cost trade-off problem in construction projects. Another contribution includes the improvement in the way the ministry of housing and other ministries select their contractors. Furthermore, the algorithm developed helps contractors to select the crew formations to

save time and cost. The efficiency of the algorithm developed is demonstrated by finding out the most suitable construction plan for the construction of 156 villas. This by itself indicates the practicality of the algorithm proposed in scheduling the repetitive construction projects for minimum time and cost.

1.6 Limitations of the Research

There are several limitations in this study in terms of its applications and accuracy. This study is applicable when the order of the units is provided or assumed. The numbers of decision variables used are limited to 30 because this number is tested in the testing problems. Therefore, the number of activities, and units used in the study are reasonable so that the total number of decision variables are below 30 variables.

Furthermore, the case studies made in this study considered certain important activities to represent the whole project. Therefore, the results obtained are not estimating the time and cost of the whole project

Other limitation is related to the data obtained from the contractors which are made based on the contractor estimations and revised by the supervisor. Therefore, the researchers have limited power to enforce the contractors to revise the data or justify them.

1.7 Terminologies of the Research

1.7.1 Abbreviations of the Research

MOCS	Multi-objective Cuckoo Search
CSA	Cuckoo Search Algorithm
CS	Cuckoo Search
SI	Swarm Intelligence
EA	Evolutionary Algorithm
BA	Bat Algorithm
FFA	Firefly Algorithm
PSO	Particle Swarm Optimization
ABC	Artificial Honeybee Colony
ACO	Ant colony optimization
NA-ACO	Non-Dominated Ant colony optimization
WM-ACO	Weighted Method Ant colony optimization
GA	Genetic algorithm
SA	Simulated Annealing

SFL	Shuffled Frog-Leaping
DE	Deferential Evolution
HS	Harmony Search
TS	Tabu Search Algorithm
GRASP	Greedy randomized adaptive search procedure
MOOP	Multi-objective optimization
LDRC	Lane Daily Rental Cost
ECI	Early Completion Incentives
LDC	Liquidated Damages Cost
DLD	Daily Liquidated Damages
CP	Constrained Programming
ZDT1	Zitzler, Deb, and Thiele Function1
SCH	Schaffer's Equation

1.7.2 Symbols of the Research

k	Number of Eggs laid in one nest by a Cuckoo Bird

J	Number of Activities
J	Number of Units
d	Number of Decision Variables
β	Step Size Parameter
Pa	Discovering Probability
DC	Project Direct Cost
IC	Project Indirect Cost
D	Duration
$MC_{i,j}$	Material Cost of activity (i) in repetitive unit (j)
$LC_{i,j}$	Labour Cost of activity (i) in repetitive unit (j)
$EC_{i,j}$	Equipment Cost of activity (i) in repetitive unit (j)
$Q_{i,j}$	Quantity of materials in activity (i) in repetitive unit (j)
E_i^n	Cost rate of equipment in activity (i) crew formation n
C_i^n	Cost rate of Labour in activity (i) crew formation n
P_i^n	Productivity in activity (i) crew formation n

$F_{crew(ij)}$	Crew Finish Time of Activity (i) in unit (j)
$d_{i,j}$	Duration of Activity (i) in unit (j)
$S_{crew(ij)}$	Crew Start Time of Activity (i) in Unit (j)
$inter_{(ij)}$	Interruption days in Activity (i) after Unit (j)
IRC	Cost of Idle Resources
$S_{i,j}$	Start Time of Activity (i) in Repetitive Unit (j)
$F_{i,j}$	Finish Time of Activity (i) in Repetitive Unit (j)
Δ_{ij}	Difference between Crew Start and Logical Start
$Shift_i$	The Activity Time Shift to Match Logic
$inter_i$	The Interruptions in Activity (i)
R	Total Interruptions in Project
LRC	Lane Rental Cost
SD	Duration Specified by Contract
DCI	Daily Completion Incentives

Chapter 2

Theoretical Framework and Previous Studies

2.1 Optimization

2.2 Meta-Heuristics in Construction Management

Chapter 2

Theoretical Framework and Previous Studies

This chapter introduces and reviews the previous research on optimization in general and repetitive construction projects in particular. In the first part of the chapter, a review is carried out on optimisation and the methods available to obtain the solutions of optimization problems. In the second part of the chapter, Meta-heuristics as optimization techniques and their application in construction management are summarized. In the third part, evolutionary algorithms and swarm intelligence based optimisation techniques are reviewed. The last part in the chapter covers cost estimation and project scheduling. In this part the review of time-cost trade-off of repetitive projects especially as multi-objective optimization is carried out.

2.1 Optimization

Optimization is the art, or science of mathematically searching and choosing best solution(s) among set of alternatives **x** called decision variable vector covered by a search space called $\Re_n$. It involves either maximizing or minimizing major function(s) $F(\mathbf{x})$ bound by h_j and g_k functions which are called inequalities and equalities constraints respectively. Optimisation is mathematically expressed in the following form:

$$\min_{x \in \Re_n} / \max \; F_i(\boldsymbol{x}) \qquad (i = 1, 2, \ldots, M) \qquad \text{Equation 1}$$

$$Subject\ to \;\; h_j(\boldsymbol{x}) \leq 0 \qquad (j = 1, 2, \ldots, J) \qquad \text{Equation 2}$$

$$g_k(\boldsymbol{x}) = 0 \qquad (k = 1, 2, \ldots, K) \qquad \text{Equation 3}$$

$$\boldsymbol{x} = (x_1, x_2, \ldots, x_n)^T \qquad \text{Equation 4}$$

2.1.1 Optimization Classifications

(Yang, 2010) explains several factors that classify optimization problem, as shown in Figure (1) and define its nature in optimization:

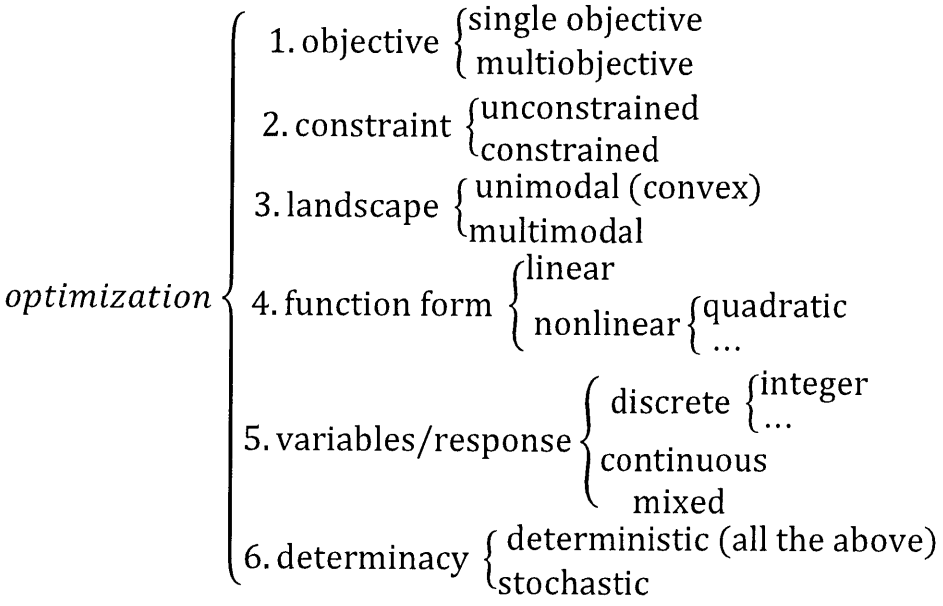


Figure (1)
Classifications of optimization (Yang, 2010)

1. Number of objectives: the first class is a single objective optimization if one objective is considered and the other one, which is more realistic, when more than one criteria or attribute, is considered in optimization called multi-objective optimization.

2. Number of equality and inequality constraints involved in the problem, that is, when all constrains are with equality it is called equality-constrained problem while if it's only constrained with inequality constraints in it is called inequality-constrained. Also if there are no constraints in the problem, it is called an unconstrained optimization.

3. Functions that form optimization problem can be linear or nonlinear functions. That is, when all the constraints are linear, the problem is called linearly constrained problem and when constraints and objective functions are linear, it is called linear programming problem.

4. Landscape of the objective functions, which refers to the number of peaks, occurs in the objective functions. When there is only unique global optimum and there is certain convexity, then the optimization problem is said to be unimodal while when there are more than one mode, it is called multimodal.

5. Type of design variables; when the variables in the optimization problem require to take integer values, it becomes an integer programming problem while if they have to be selected from a set of values which are discrete, it is called discrete optimization or combinatorial optimization. Besides, optimization is considered continuous when real values are used in some interval so when both discrete and continuous values are used, optimization is a mixed type.

6. Determinacy: Deterministic approaches take advantage of the analytical properties of the problem to generate a sequence of points that converge to a global optimal solution. Stochastic optimization refers to

the minimization (or maximization) of a function in the presence of randomness in the optimization process.

There are several mathematical programming techniques used in optimization such as the simplex method used to obtain the solution of linear programming and its extension is suitable for quadratic programming. When the optimization problem has nonlinear objective function and constraints non-linear programming techniques are required to be used to determine its solution. These methods work in an iterative manner and they require the selection of initial design point to initiate the iterations as well as the computation of gradients of objective function and constraints. Among these techniques Newton's method, quasi-Newton method, finite difference, conjugate gradient methods, Interior point methods, gradient descent, sub-gradient methods, bundle method of descent, ellipsoid method, reduced gradient method (Frank–Wolfe) can be mentioned. Simultaneous perturbation stochastic approximation (SPSA) method for stochastic optimization makes use of efficient gradient approximation. Alternative methods to determine the solution of optimization problem are heuristics and meta-heuristics which use random numbers in searching the solution domain for the optimum solution. These methods are widely used in providing optimum solutions for optimisation problems in engineering design and management.

2.1.2 Meta-heuristic Algorithms

Algorithms are like plans used to find a treasure in a land within a limited time so that it is not reasonable to search in every square meter but to do random walks to make hints. Therefore, optimization does random selection just like that of a blind-folded person –"which is the

main characteristic of modern search algorithms". Also, it is combined with confident selection of a group of sighted people sharing the information about the treasure found in the way called swarm intelligence. Hunters also are improved in abilities in finding the treasure just like how search agents are improving in evolutionary algorithms and that only the best solutions or agents are kept to continue the search process.

That is to say, there are two different categories of optimization algorithms perform optimization as shown in Figure (2):

1. Deterministic algorithms are those algorithms that if starts at a certain point it will keep repeating the same path and values of both design variables and the functions when performing it any time just like the simplex method in linear programming. Furthermore, when gradient information used in the program it is called "gradient-based algorithms" just like Newton-Raphson algorithm. On the other hand, when it does not use any derivative it is called gradient-free algorithms it such as Hooke-Jeeves pattern search and Nelder-Mead downhill simplex. (Yang, 2010)
2. Stochastic algorithms have randomness that the path and values changes in each time the program is run. In addition, when mixing both deterministic and stochastic algorithms the result is a hybrid program which is also considered as a type of stochastic algorithm due to the random component it contains. There are general two types of stochastic, heuristics - used for simply discovering solutions by trial and error- ,

and meta-heuristic for higher level of search i.e. better on global optimization. (Yang, 2010)

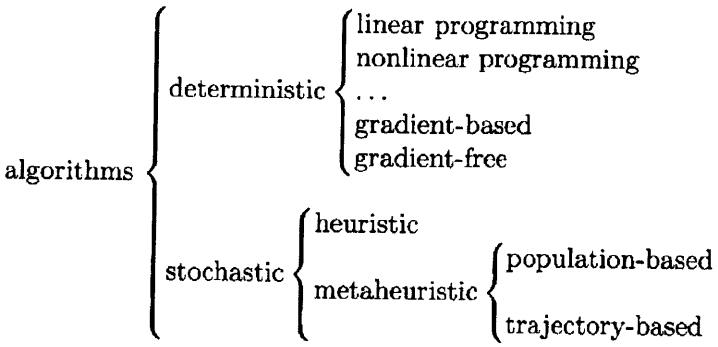


Figure (2)
Classification of algorithms

The word Meta-heuristics is made of two parts derived from Greek terms, which are the suffix "meta" and the word heuristics. Meta means beyond, in an upper level and the word heuristic derived from the Greek verb heuriskein that means ''to find'' a good solution from the solution space by the means of trial and error. When combining the two terms together, the definition of Meta-heuristics is clearly the higher level of heuristics. Furthermore, the definition Meta-heuristics are advanced research procedures, strategies or methodologies (templates) that provide acceptable solutions in a reasonable time. During the last few decades, they are designed with algorithmic framework to underlie the nature to more successfully solve complex optimization problems with uncertain, stochastic, and dynamic information (Talbi, 2009) (Bianchi et. al, 2008). Therefore, meta-heuristics are simply described as strategies converted into algorithms that are derived and inspired from natural phenomena. Metaheuristics are classified into different groups based on the way it search for solutions, as shown in Figure (3). Meta-heuristics explore

solution space to recognize regions of higher quality solutions using strategies to deliberately select between intensification and diversification (Bianchi , 2008). Due to its diversity, they started emerging as a major player for multi-objective global optimization, as they often mimic the successful characteristics in nature, especially biological systems (Yang, 2011).

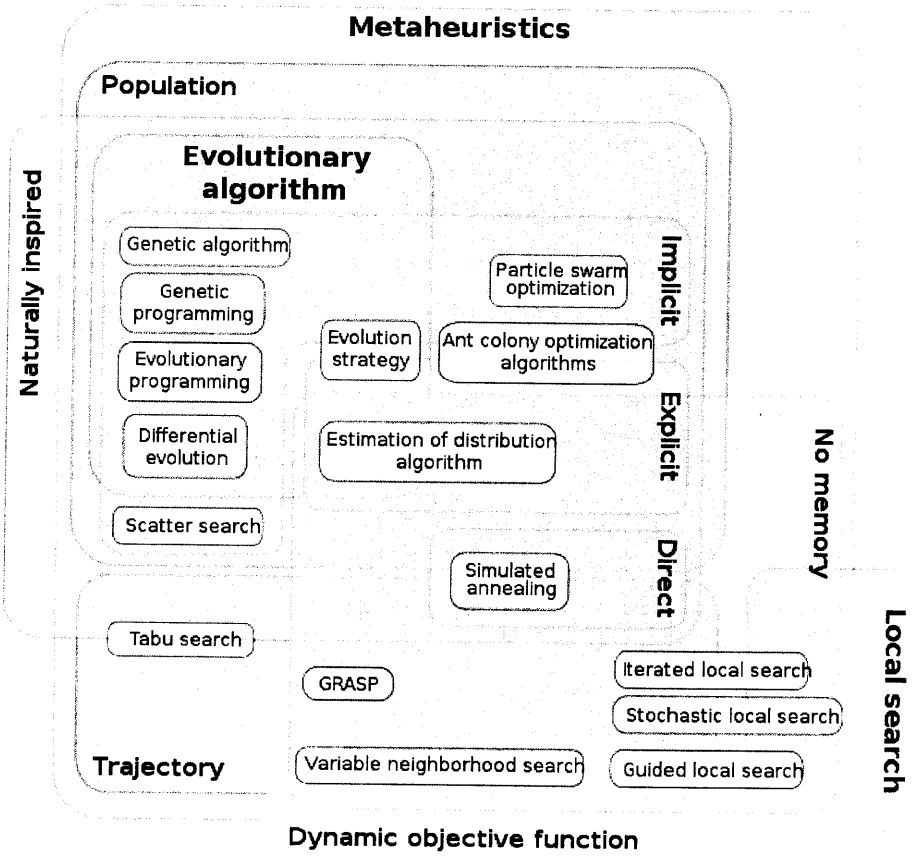


Figure (3)
Classifications of Meta-heuristics (Dréo, 2011)

The following steps are required to be followed before meta-heuristics are applied to determine the optimum solution of an optimization problem (Talbi, 2009) .

1. The formulation of problem by identifying it to create the initial statement including the internal and external factors and objective(s).
2. Model the problem by building the mathematical model to reduce the problem to well-studied optimization models which are considered as simplifications of the reality.
3. Optimize the modelled problem to produce good quality solution for the problem and present the solution. Then when taking constraints into account, solutions that violate them are considered to be infeasible.
4. Implement solution to check whether solutions are acceptable in practice. This is needed to improve the model since meta-heuristics need continuous improvement since they are developed. This will lead to find better fitting for some advanced approaches like multi-objectives optimisation.

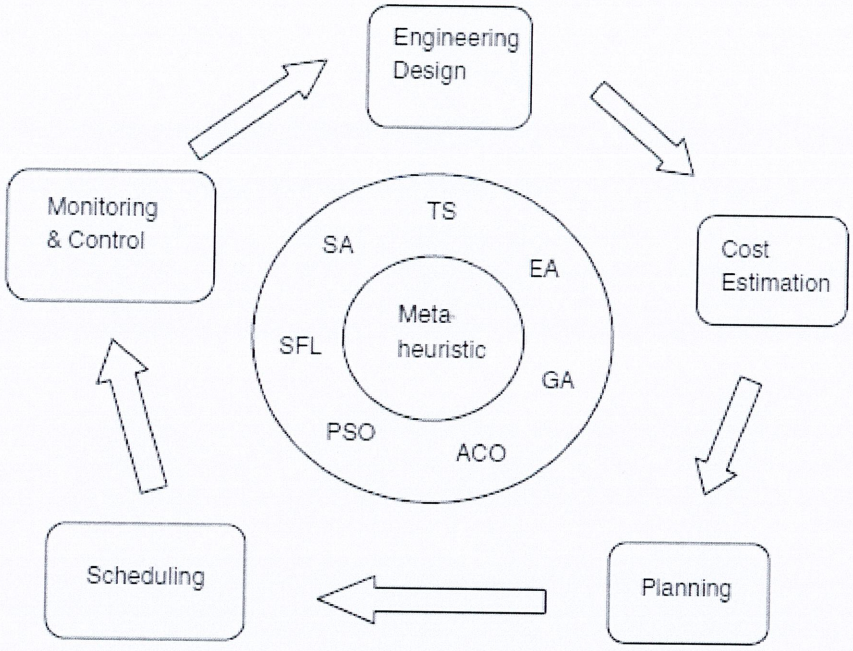


Figure (4)
Meta-heuristics used to optimize construction management objectives (Liao et. al, 2011)

Figure (4) states different stages in construction project life cycles where the application of the meta-heuristics is common. It is demonstrated that they are better in computation time comparing to exact methods (mathematical programming techniques) especially in large scale problems. The meta-heuristic algorithms shown in Figure (4), namely, simulated annealing (SA), ant colony optimization (ACO), evolutionary algorithm (EA), genetic algorithm (GA), particle swarm optimization (PSO), and shuffled frog-leaping (SFL) are inspired by the natural surroundings. (Liaoet. al, 2011). These meta-heuristic algorithms simulate natural features and they are shown to be efficient by (Yang and Deb, 2010). These algorithms have two characteristics which are intensification and diversification. Intensification stands for the ability of meta-heuristics to use best solutions as a reference in their search and

diversification stands for the randomized solutions which allow for more diverse solutions. (Yang and Deb, 2010).

Brief History of meta-heuristic algorithms

The history of meta-heuristic algorithms is given briefly in (Yang, 2010). Among these the GA was basically inspired from Darwin's theory and based mainly on crossovers and mutations to generate solutions. It was very widely applied in modelling and finding the optimum solutions of engineering design as well as engineering management problems. It is later progressed in 1975 to cover even noisy, multimodal and discontinuous systems.

The most important time in optimization was in the 1980's and 1990's as more algorithms appeared with natural inspirations. In the 80's SA -inspired by the annealing process of metals- was developed to be an optimization algorithm. Likewise, in the 90's, important algorithms appeared like the (ACO), (PSO), and Differential Evolution (DE). In the 21 century, the modernization continued with several engineering problems solved after the development of Harmony Search (HS). In 2005, Artificial Bee Colony was developed by D. Karaboga. After that, in 2008 Xin-She Yang developed the Firefly Algorithm (FA), followed in the next year by Cuckoo Search Algorithm (CS) and then Bat Algorithm (BA) in 2010. (Yang, 2010).

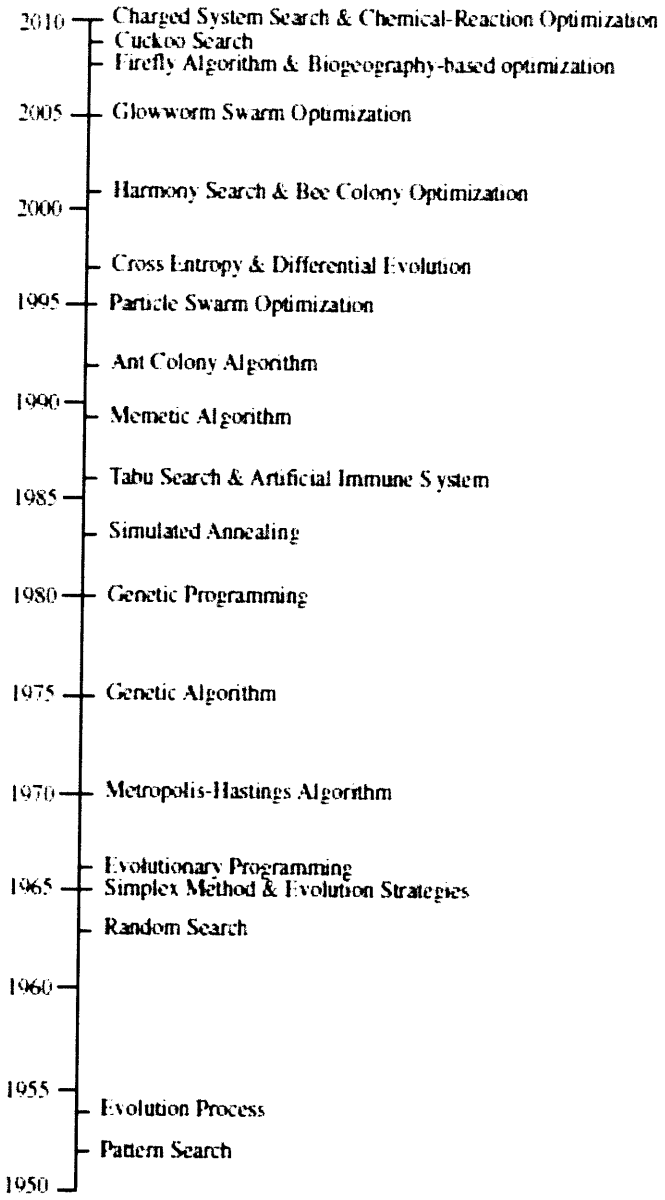


Figure (5)
Timeline of meta-heuristics (Gandomi et. al, 2011)

Evolutionary Algorithms

Evolutionary algorithms are algorithms inspired and developed from the Charles Darwin's theory in 1859. The theory developed into strategies, programing and approaches that mainly aim to solve complex problems such as optimization in vital domains including engineering

designs. The common term in evolutionary algorithms is the population as shown in Figure (6) to create a tentative solution. The algorithm is made to perform iterations each corresponds to a generation (Dorigo et. al, 2008). Therefore, when the fitness of an objective function indicates better quality, the populations will be considered to be the parents in reproduction. There are several means of the reproduction that aim to form a variation from the parents to the children such as crossover or recombination. Repeatedly, the best qualities off-springs will replace the worse to maintain ensure high quality solutions survival. (Talbi, 2009)

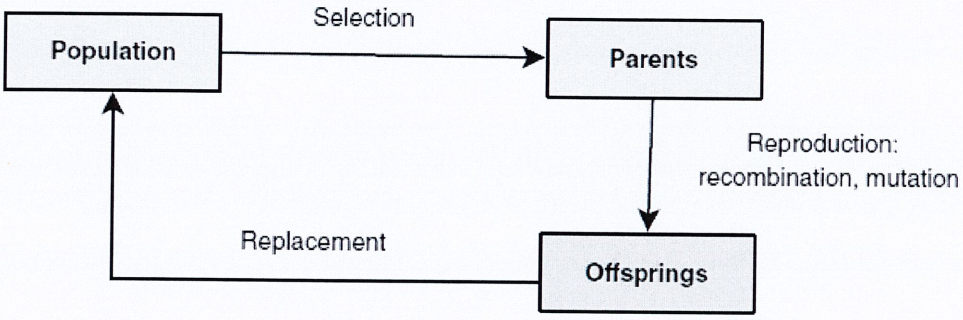


Figure (6)

One generation in any evolutionary algorithm (Talbi, 2009)

The general template for evolutionary algorithm is shown in Figure (7). There are three types of evolutionary algorithm which are "Evolutionary Programming (EP), Evolutionary Strategies (ES), and GAs (Marco et. al, 2008). There are also several implementations such as genetic programming, gene expression programming, differential evolution, neuro-evolution, and learning classifier system. (Dréo, 2011). Recently, the developments in artificial intelligence and computer technology outweighed the non-traditional optimization technique -such

as GAs which imitates the natural evolution-in searching complex solution spaces. (Hegazy and Wassef, 2001).

There are some advantages explained by (Rajabioun, 2011) for using the evolutionary algorithms:

1. Its robustness to dynamisms as they can be used to adjust solutions to changing conditions.
2. Broad applicability in a wide range of problems.
3. Hybridization with other traditional methods and techniques.
4. Solves problems that have no solutions and when there is no human expertise.

```
Generate(P(0)) ; /* Initial population */
t = 0 ;
While not Termination_Criterion(P(t)) Do
   Evaluate(P(t)) ;
   P'(t)      = Selection(P(t)) ;
   P'(t)      = Reproduction(P'(t)); Evaluate(P'(t)) ;
   P(t + 1)   = Replace(P(t), P'(t)) ;
   t = t + 1 ;
End While
Output Best individual or best population found.
```

Figure (7)
Evolutionary algorithm template (Talbi, 2009)

Genetic Algorithm (GA)

This is the most popular type of evolutionary algorithm that mimic the natural evolution to find solutions to problems using natural selections such as inheritance, mutation, selection, and crossover. The algorithm generates random initial solutions (termed phenotypes, or individuals) within the possible search space in the first population as a basis for the

next generation, and then keeps regenerating new solutions iteratively. That is, when the form of solutions is in numbers such as the binary (string of 0's and 1's) is aimed to regenerate into new better solutions; the algorithm uses at least one of the operators known as recombination/ crossover operator and mutation operator as well other possible operators like regrouping, colonization-extinction, migration, mating, selecting, etc. (Akbaria and Ziaratia, 2010) Technically, the algorithm iteratively applies fitness of the objective function to evaluate the quality of solutions in order to keep the most of the high fit to improve and little less fit to ensure diversity. This is due to the fact that GA uses at least one of the key production tools (called operators) in order to "breed" a new "child" solution from two old high quality "parent" solutions. Specifically, the biologically inspired operator' i.e. crossover /recombination and mutation allow child solution preserving the features of its high quality "parents". Moreover, this metaphor is clearer when using the term "chromosomes" to indicate the solution variables.

Reproduction in evolutionary algorithm is the process of selecting individuals to form the parents to produce new children which is in two complementary ways mutation and crossover. That is, crossover carries characteristics from parents to offspring while mutation creates diversification in current individuals to create potential missing values.

Mutation Reproduction

The mutation process causes small changes in the chosen individuals based on probability of effecting one gene. The probability ranges from [0.001 to 0. 01] or simply set to 1/d where d is the number of decision variables. It worth pointing that mutation probability is significant

because very small mutation rate causes drift or dissolved genetic variations in next generations and causes loss of good solutions if too high but if elitist selection exists. However, the mutation needs enhancement to cover the search space, validity of produced solution and locality to produce a minimal change. Ultimately, the mutation is linked to neighbourhood structure representations such as binary representation or flip operator, discrete representation or switching a value with another from the alphabet, and permutations that is based on the swapping, inversion, or the insertion operators. (Talbi, 2009)

Crossover Reproduction

Recombination or Crossover is a binary or n-array operator that makes use of two parents' chromosomes to produce children. This procedure takes place with consideration of heritability from parents and validity of offspring. The rate of crossover ranges from 0 to 1 which is the proportion of parents operates but the frequently used rates range from 0.45 to 0.95. There are representations for crossover which are basically proposed for binary is either n-point or uniform distribution to generate two off-springs with equal interchanged segments from parents. The n-point creates n random crossing sites while the uniform crossover uses uniform distribution to select the crossover point k regardless the size of segments as shown in Figure (8).

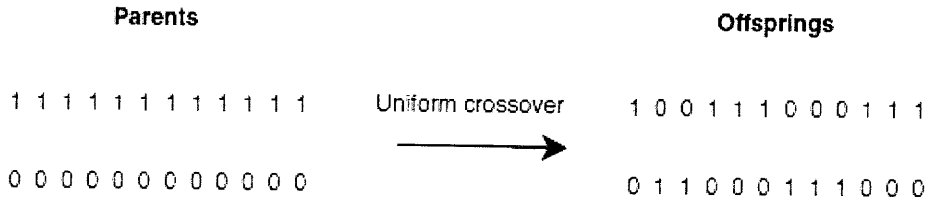


Figure (8)
Normal crossover (Talbi, 2009)

Swarm Intelligence Based Optimization Algorithms (SI)

These algorithms are mainly inspired by the behaviour of unsophisticated biological agents like bacteria, insects, birds and fish. These agents are simple as per individual, as they are self-organized in changing their surroundings though they are cooperating in swarms by communication. These agents have "intelligent" global patterns although they are not controlled by any centralized system over their behaviour. These algorithms involve decentralized populations that represent the solutions. The algorithms use randomization for their self-organization and considerations for local interaction which will indeed emerge to optimum global pattern. Each agent symbolizes a potential solution for the optimization problem that will move in position when it shows improvement in solution per consecutive iteration. Examples for these algorithms include (ACO), (PSO), Artificial Bee Colony, Firefly Algorithm (FA), (CS), and the Bat Algorithm (BA). These algorithms revealed overweighing results over traditional algorithms due to their simplicity, robustness, and diversity of solutions without need for centralized control of a global model. These algorithms have ability to find solutions of complex nonlinear programming problems and found increasing number of applications in publications. (Saka et. al, 2013)

Two important means of exploring the solutions used in the SI based algorithms:

1. Exploitation: means to generate new better solutions based on previous solutions to let the solutions convergence well though it could be trapped in a local optimum.

2. Exploration: means to explore the search space further to generate diverse solutions far from the current solutions to find a global optimum although this might cause the slow convergence and waste of computational efforts.

Ant Colony Optimization (ACO) Algorithm

ACO algorithm is developed by Dorigo et al. (1991), Dorigo (1992), and Dorigo and Stu¨tzle (2004) inspired by the intelligence of ant in colonies. The idea of Ant colony optimization (ACO) came from the way ant colonies find the path to food source based on results of studies conducted on the managerial prospective of ants. The idea of swarm intelligence is clear in this algorithm as the ants are completely blind, but very efficient in discovering shortest route that takes them from their nest to the food source. The results of studies regarding ant colonies showed that ants deposit volatile substance -called pheromones- on the ground they travel to let other ants sense it by antennas. Therefore, when the ant colonies find more concentration of pheromone on a route they understand that it was passed by previous ants and thus a great majority will travel on this route.

The steps of ACO are as follows: (Saka et. al, 2013)

1. Assigning ants for the potential solutions to the optimization problem, defining the pool for decision variables, and selecting random ants for each decision variable in order to calculate initial pheromone amount from the minimum value of the objective function.

2. Each ant in the colony selects values from the pool for decision variable under probability with considerations to local trail values and visibility.

3. Ants will decide upon the path based on the concentration of pheromone bearing in mind a lower pool to encourage them to keep searching and to simulate the natural evaporation of the pheromone.

4. Keep assigning a value from the pool to the rest of decision variable until all the ants in the colony takes a value for each decision variable.

5. All above are describing one cycle, so to start another one, all ants will coming back to their first decision variables and the rest of steps will reiterate.

6. When stopping criteria is not yet met, the algorithm will keep repeating the previous steps, common example of stopping criteria includes a fixed number of iterations.

Particle Swarm Optimizer PSO

This algorithm is inspired by the social behaviour of animals such as fish schooling, insect swarming, and birds flocking. There are four rules - after they were amended- in defining the behaviours. (Saka et. al, 2013)

1. Each agent is attracted toward the location of the roost;
2. Each agent remembers where it was closer to the roost
3. Each agent shares information with others about its closest location to the roost.
4. The PSO picks the number of particles (population size) as possible solutions to represent one swarm and then controls the changes in particles' velocities by their maximal change of a particle velocity.

Therefore, there are three main factors describes the movement of the particles by which the swarm gain its cumulative experience: position, velocity, and fitness. The position refers to the decision variables of the problem, the velocity is the speed allowed to change the position, and the fitness is the quality or value of the objective function at each position. Accordingly, the optimization using PSO is summarized in the following three concepts:

1. Optimization problem starts with random initial position (solution), initial velocity and fitness value then it will move in the search space, but keep in mind the position of the best solution(s).
2. Each particle moves with weighted sum of three components of velocities which are: the old velocity, a velocity component

that drives the particle toward the location in the search space where it was previously best solution, and a velocity component that drives the neighbour particle toward the location in the search space to the PS algorithm.

3. Each time the PSO will update the optimum particle position velocity vector of each particle and fitness. (Saka et. al, 2013)

Artificial Honeybee Algorithm ABC

The ABC algorithm stimulates the intelligent behaviour of a honeybee colony which is divided into three groups of carrying different responsibilities:

1. The first group is called the "employed bees" and their job is to fly away from the hive to locate and remember the food source, and evaluate the quantity of nectar. Consequently, when they come back they will share this information with other bees by dancing in the dancing area expressing larger quantities of nectar by the longer dancing time.
2. The second group is the onlooker bees who perceive the information and decide whether to fly to the food source if it worth.
3. The third group is scout bees which carry out the random exploration of the area around the hive to find new locations of food sources.

The algorithm uses food resource as potential solution and the nectar amount found in it as the quality/fitness of the solution. The ABC algorithm consists of the repetitions of the following four stages:

1. Initialization of population of good sources phase, and population size representing the number of artificial bees
2. Employed bees phase: find random food sources, calculating the fitness, and replace the previous solution with it if the new one is better.
3. Onlooker bees' phase: they choose their food source probabilistically related to the fitness values of each food source in the population.
4. Scout bees phase: they abandon the solutions when it cannot be improved after particular number of trials. (Saka et. al, 2013)

Firefly Algorithm FFA

This algorithm mimics the behaviour of fireflies and their flying patterns when they communicate, search for prey, and find mates using bioluminescence. The algorithm makes use of three rules as follows:

1. All fireflies are unisex so they attract each other.
2. Attractiveness is proportionate to brightness of the firefly, so less bright one will be left away. Moreover, both of attractiveness and brightness are inversely proportional to distance.
3. The brightness of a firefly is determined by the landscape of the objective function.

There are several simple steps to apply FFA as given below:

1. Producing random initial population (solutions) of fireflies

2. Calculating light intensity and distance between fireflies
3. Moving a firefly toward brighter fireflies but if there is other brighter firefly then the moving will be random.
4. Evaluating the new solutions and update light intensity.
5. Rank the fireflies and find the best solution so far.

(Saka et. al, 2013)

Cuckoo Search Algorithm CSA

This Algorithm simulates the reproduction strategy of the cuckoo birds which lay their eggs in the nests of other birds. When the other bird discovers that the egg is for a cuckoo bird, it will expel the egg from the nest or sometimes demolish the nest. Therefore, the cuckoo birds use clever ways to not give a chance to the other bird to discover the foreign eggs. Sometimes, the cuckoo replaces the existing eggs in the nest with theirs, and for some species, the cuckoo mimic the pattern and colour of the eggs of host birds. (Saka et. al, 2013). This algorithm is explained in more details in Chapter 4.

Bat Algorithm BA

This algorithm simulates the micro bats which use sonar called echolocation to find their prey, prevent obstacles, and to trace their roosting crevices. They produce a very loud sound pulse and notice the bounded back echo by their surroundings. Their pulses vary in properties and can be correlated with their hunting strategies, depending on the species. They make clear three-dimensional acoustic image in mind using the delay time between emission and the detection of echo, time

difference between two ears, and loudness variations of the echoes. This three-dimensional image includes the distance, orientation, type and size of prey, and the speed the pray is moving with. These echolocation capabilities can be interpreted algorithmically when considering the following points:

1. All bats use echolocation to supernaturally detect the distance, and distinguish between their target and its background barriers.

2. Bats search for their prey by flying randomly with a velocity of a constant frequency, varying wavelength, and loudness. They adjust the frequency of the pulses and its rate of emission based on the nearness of their target.

3. The loudness is presumed to fluctuate from a constant maximum value to a minimum value

The repeating steps of the algorithm that practice the previous theoretical implication are as follows:

1. Initializing the bat population with certain position and velocity to symbolize a solution as well as pulse rates, loudness and frequency

2. Computing the new solutions and velocities based on uniform distribution driven random numbers and finding the current global best (solution).

3. When a randomly generated number is larger than the loudness, finds a solution from the best solutions.

4. Producing new solution in the means of a local random walk.

5. If the generated random number is larger than the initial loudness and the new solution dominates the previous, then reduce the initial loudness and increase the initial pulse rate.

(Saka et. al, 2013)

2.2 Meta-Heuristics in Construction Management

Construction management involves planning, coordination, and control of construction projects to satisfy an owner requirement in terms of three constraints which are time, cost and quality. The construction projects ranges from private projects to governmental projects and in different civil engineering works like buildings, bridges, highways, and other infrastructures. These projects naturally require cost estimation and scheduling of activities whether in the planning stage or execution stage.

2.2.1 Cost Estimation in Construction Projects

Cost estimation is one of the vital topics and concerns in construction management which predicts the approximate cost of a project and required operations. There are several levels of cost estimations that range from rough estimations of whole units to detailed estimation of components and operations. There are early estimates which are important to clarify the cost of construction projects in terms of early decisions in feasibility regarding the budget. Therefore, an estimator needs clear and specific data about the tasks uncertainties as well as inflation and extra expected costs. On the other hand estimating a price of a construction project is made by calculating the total potential costs

together with the sought profit. Therefore, contractor defines cost rates for being able to estimate costs for bids and later to plan the construction project.

2.2.2 Project Scheduling

Scheduling problems are concerned with estimating start and finish times of activities in a project. There are well-known techniques for scheduling such as Program Evaluation and Review Technique (PERT) and Critical Path Method (CPM). Those two methods are only used when the "project deadline is not fixed and the resources are not constrained by either availability or time". One of the models is used by (Liao et. al; 2011) to trade-off time-cost and smooth the resource demand. The scheduling model in repetitive construction projects produces a resource-driven schedule for repetitive units in the project that fulfils the "logical precedence relationships, crew availability and crew work continuity requirements". (Hyari et. al, 2009)

2.2.3 Time-Cost Trade-Off with Meta-Heuristics

One of the important features of project management is the time-cost trade-off because it is concerned with two important constraints that conflict in nature, time and cost. For instance the more expensive crew formation used, the quicker they have to complete the activity. In construction projects the crashing rate curve is used to determine intermediary points between two extremes which are the minimum cost and the minimum duration. For example, minimum costs are assigned for non-critical activities and called the "normal cost" while the minimum duration or "crash time" is assigned for critical activities and costs called "crash cost". Another solution is to compress the project activities to

reduce the project duration to meet the finishing time. To optimize cost-time trade-off, the problem could be defined as single objective or multi-objective optimization. When the problem is defined as a single objective time-cost trade-off objective it becomes minimizing the cost of compressing activities, minimizing the crash times or optimizing a linear time cost correlation using a management system. One of the application is reported by (Liao et. al, 2011) where 18-activity construction project is optimized in order to meet the deadline at the minimum total cost considering "five optional methods of construction varying from cheap-and-slow one to fast-and-expensive one". One other applications is carried out by (Tang et. al, 2014) who used linear scheduling and constraint programming considering unforeseen circumstances to control schedule made by a scheduling model. The results from scheduling model and optimization algorithm validated the actual data taken from a real highway construction project. However, they claim that traditional "network planning methods or the Gantt schedule" is inadequate to solve linear scheduling.

On the other hand, when multi-objective time-cost trade-off is used, it minimizes both project duration and cost in a Pareto curve. Meta-heuristics has several vital applications in construction management topics specifically in finding optimum solution(s) to certain objective function. According to the review made by (Liao et. al, 2011) evolutionary algorithms (GAs) are widely used in construction management among all because they are the early techniques and capable to deal with both continuous and combinatorial optimization problems that allows to find good solutions. Furthermore, when two objectives are essential in decision making and that are conflicting in nature, they are

optimized as multiple objectives and sometimes as a single weighted objective. There are number of applications of optimization in construction management mentioned by (Liao et. al, 2011) as shown in Figure (9). ACO, EA/GA, HS, and SFL algorithms are applied to time cost trade-off problems in optimisation. The number of papers reviewed were 12 papers, 8 of them were in multi-objective algorithms which makes it the most frequently studied application.

Topic	SA	TS	ACO	EA/GA	PSO	HS	SFL	Hybrid	More than 1
3. Engineering Design				5					
4. Cost Estimation				3					
5. Planning									
5.1. Site preparation	1								
5.2. Site/floor layout		1	1	9	1			1[3]	
5.3. Site routing				1					
5.4. 1D stocking cutting				2					
5.5. Supply chain/logistics			1	1					
5.6. Equipment selection				3(1)					
6. Scheduling									
6.1. Resource unconstrained	2		3	2					
6.2. Resource constrained									1[2]
6.2.1. Time-cost tradeoff			3(3)	12(8)		1(1)	1		
6.2.2. Resource allocation			2(1)	8(3)	2(1)				
6.2.3. Resource leveling	1			3					
6.2.4. Integrated models				5(2)					
7. Monitoring and Control				1					

Note: (1) Numbers in parenthesis indicate number of papers that consider multiple objectives.
(2) This study employs two metaheuristics, i.e., SA and TS.
(3) This is a ant-GA hybrid.

Figure (9)
Meta-heuristics used in solving construction projects problems in literature (Liao et. al, (2011)

Repetitive activity (i)	Excavation (i = 1)				Foundation (i = 2)				Columns (i = 3)				Beams (i = 4)				Slabs (i = 5)			
Repetitive unit (j)	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4
Quantity of work (Q_i^j) in m^3	1147	1434	994	1529	1032	1077	943	898	104	86	129	100	85	92	101	80	0	138	114	145
Available crew options																				
Crew formation (n_i)	1				1	2	3		1	2	3		1	2	3	4	1	2		
Productivity (P_i^n) in m^3/day	91.75				89.77	71.81	53.86		5.73	6.88	8.03		9.9	8.49	7.07	5.66	8.73	7.76		
Project cost Data																				
Labour cost in $/day	340				3804	2853	1902		1875	2438	3000		3931	3238	2544	1850	2230	1878		
Equipment cost in $/day	566				874	655	436		285	371	456		315	259	204	148	177	149		
Material cost in $/$m^3$	0				92				479				195				186			

Figure (10)

Quantities of work, available crew formation options and their related costs and productivities (Hyari et. al, 2009)

2.2.3 Previous Studies in Multi-Objective Cost-Time Trade-Off Problems

The study of (Hegazy and Wassef, 2001) proposed a non-traditional trade-off for scheduling and total cost optimization that finds combination of crew formations, number of crews, and interruptions. It aimed to develop robust and consistent spread sheet, with macro-programs and incorporates benefits of CPM and line of balance techniques that confirms work continuity for non-serial activities.

The study by (Hyari and El-Rayes, 2006) presents a planning and scheduling of repetitive construction projects as a multi-objective optimization model. It basically aims at minimizing the duration and interruptions -maximizing continuity- It uses an example of varying options of crew formations each with productivity level shown in Figure (10). The paper considered three span concrete bridges to find best crew formations and interruption days. These three spans are consisting of four repeating units with different quantities as shown in Figure (10). It is also clear in the figure that the activities considered in this project are excavation, foundation, columns, beams, and slabs. One activity is abstracted from a whole schedule show a sample of calculation shown in Figure (11). A broad proposal is for (Senouci and Naji, 2006) who presented a system of GA optimizing scheduling and total cost of both serial and non-serial linear projects. The optimization is made to determine optimum crew formations and interruption durations of the production activities.

(Liu et. al, 2007) presented a flexible and constrained programming (CP) model for optimizing the total cost or project duration of linear

projects including non-repetitive activities. There are other models used in the field of optimizing of time and cost.

Later, the case study of (Hyari and El-Rayes, 2006) was extended to be in optimizing for project duration and total cost by (Hyari et. al, 2009). They proposed an automated model that support planning and scheduling optimization for repetitive construction projects. The model performed multi-objective optimization of time and direct cost using GA approach to get a set of trade-off solutions. This involved four modules which are "scheduling, direct cost, optimization and total cost modules"` as shown in Figure (12). The cost model is used after optimization to add indirect costs and penalties due to delay behind completion date or incentives due to finishing earlier than completion date in order to find one optimum solution with minimum cost.

The model made by (Afshar et. al, 2009) applied three different heuristics techniques to trade-off time–cost problems of 18 activities shown in Figure (13). They developed multi-colony ant algorithm to solve the time–cost problem in means of weighted method multi-objective optimization problem. Besides, there are other papers and researches considered cost and time trade-off but considering specific types of projects. One of them is for (Ezeldin and Soliman, 2009) who proposed a successful hybrid combination of GA together with dynamic programming to optimize time-cost trade-off of a 14 activity repetitive non-serial mini-project.

Any of the optimization methods and approaches is ultimately used to aid decision makers to evaluate and take decisions. For that, further

binding with more models that best utilizes the solution and add more features is a useful achievement. One of the comprehensive studies which was made in Egypt was for (Marzouk et. al, 2010) which presented a framework of models. This framework included network builder, a crew formation selection, a simulation, optimization and a reporting. This was made to help authorities and construction firms' in estimation and optimization of time and cost using GA. The study mainly supports defining options that influence the duration and cost to support evaluation by answering (what if) questions.

Fan et. al, 2011) used GA optimization model for planning repetitive projects and minimizing overall cost. The model optimizes the sequence of work "soft logic" considering the impact of acceleration strategies such as increased crew size, overtime policies, or additional shifts. Finally, (Zhang et. al, 2013) proposed GA optimization for the same case study of (Hyari et. al, 2009) used for repetitive project considering varying requirements for resource continuity of activities. The model showed better results in the example used in literature particularly for durations less than 117 days shown in Figure (14).

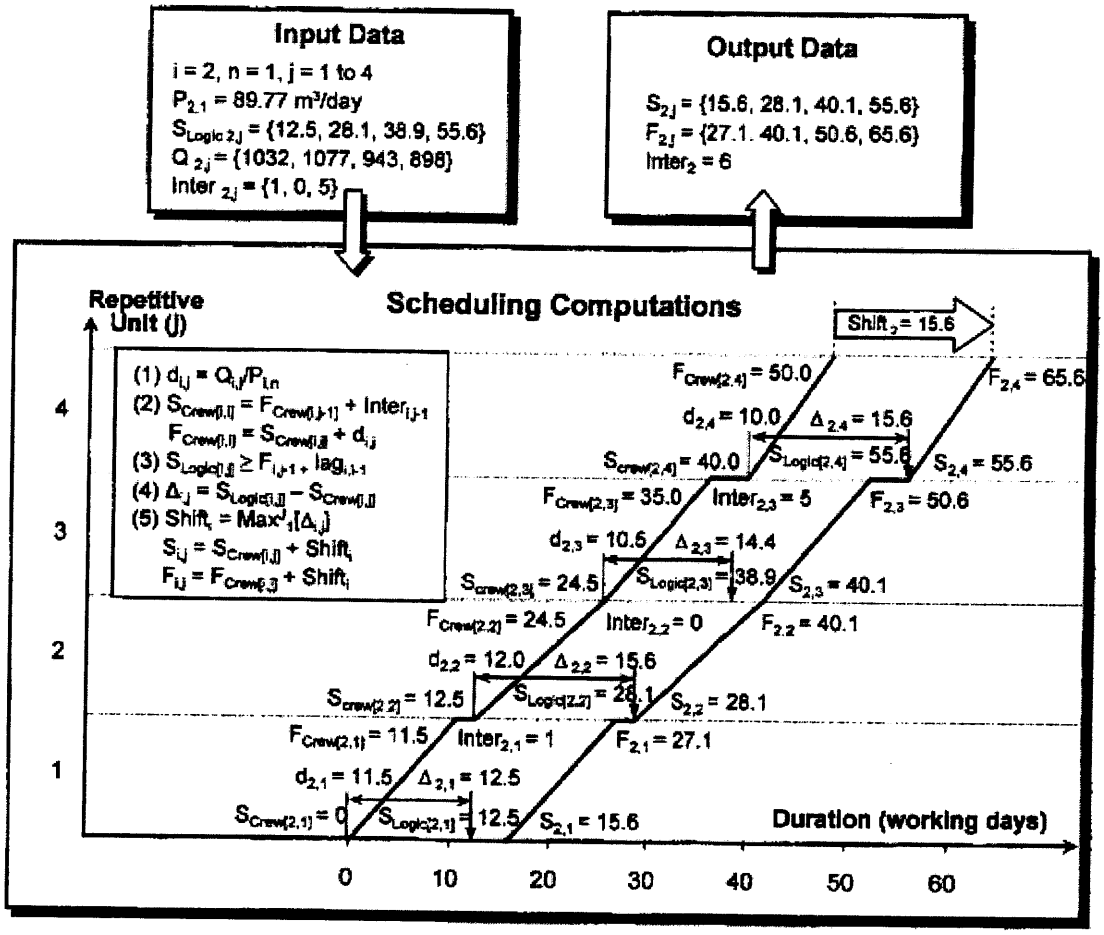


Figure (11)
Scheduling model calculation for foundation activity (Hyari and El-Rayes, 2006)

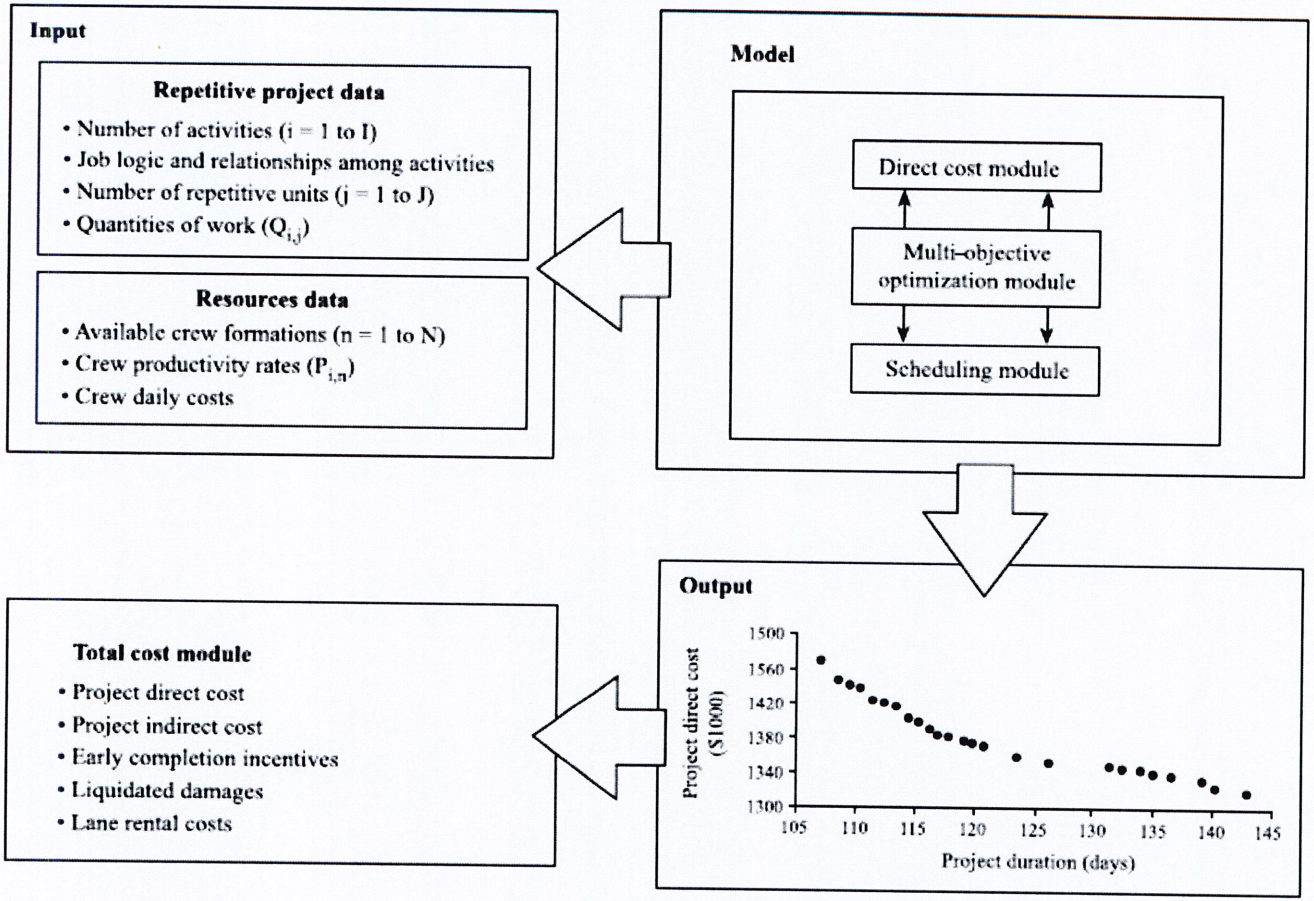


Figure (12)
Hyari's model development (Hyari et. al, 2009)

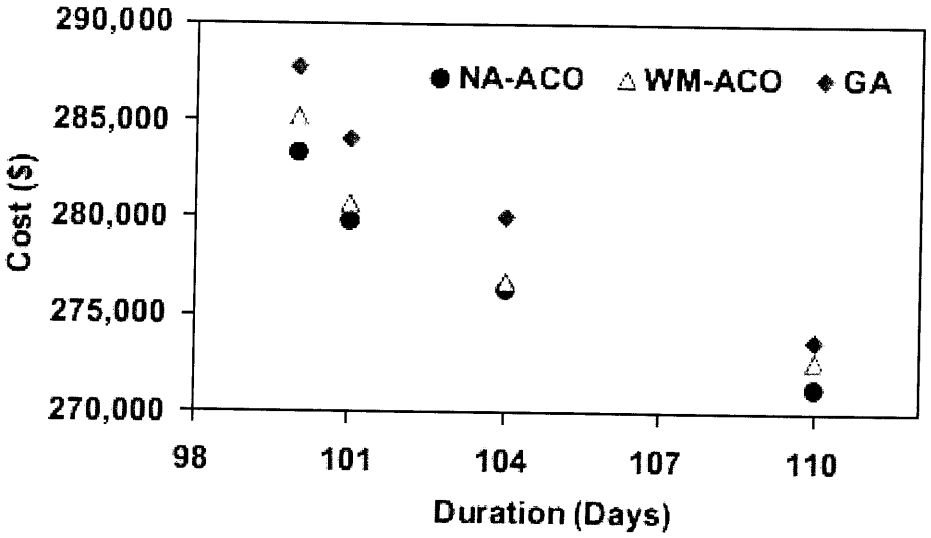


Figure (13)
Comparison between three Pareto solutions (Afshar et. al, 2009)

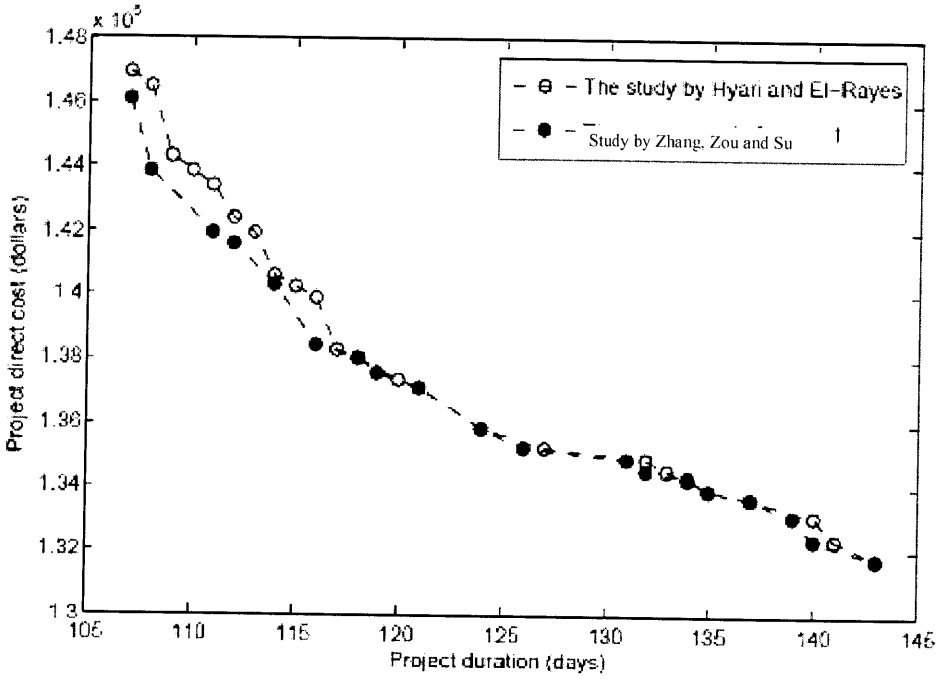


Figure (14)
Comparison of Pareto curves by different models

Summary:

All in all, the repetitive units' time-cost trade-off is rich of applications in the field construction projects. There are several models and algorithms used in multi-objective optimization. These include GA, multi-colony ant algorithm by weighted sum, heuristic techniques, spread sheet with macro, etc. These approaches are used to solve problems in varying forms. Examples include, time and direct cost trade-off of provided sequence, optimizing the sequence of work considering acceleration strategies, and continuity of serial and non-serial activities.

Benefits of these studies

These studies drag attention to the different applications in the field of repetitive construction projects. They are well supporting decision makers adapt proper techniques that saves time and cost in long term projects. These techniques are very vital in their applications and serves different purposes. One of the purposes is to help contractors choose appropriate methods in repeating projects that saves time and cost. Another application is to support clients predict best crew formations and chose contractors who will save time and cost. Furthermore, the field is also vital from the research prospective when testing optimization methods to best solve the repetitive construction projects. Specifically, this study applies the CSA to trade-off time and cost to obtain better solutions. CSA is selected in optimisation after its recognized efficiency in terms of accuracy and time consumption solving problems in previous researches. This efficiency is due to the use of less number of parameters than that used in other algorithms.

Chapter 3

Methods and Procedures

3.1 Problem Formulation and Modelling

3.2 Solution Approach (MOCS)

3.3 Experimentations and Results

3.4 Discussion and Conclusion

Chapter 3

Methods and Procedures

3.1 Problem Formulation and Modelling

A repetitive construction project consists of fixed number of activities repeated over segments, units, or locations. The challenge in planning repetitive projects is to keep resources continuity so that the project is "uninterrupted and unchanged" from start to end. Therefore, the objective functions are affected by the choice of any resource/crew formation that runs an activity over all the repeating units. One of the studies carried out to optimise scheduling of repetitive construction project is by (Hyari et. al, 2009).

The decision variables are the available crew formations used in each activity. The crew formations are defined by the productivity, labour cost rate and equipment cost rate. The productivity data are used to find the duration of each activity in each unit. Other decision variables include the interruption occur in activities after completing each unit. These interruptions occur due to the labour waiting to work in a unit until the previous activity to finish. The objective functions are two, the first find the total duration of project by scheduling the activities and units in series. The second objective is the direct cost of the project which is related to the duration of the activities per units. The constraints used in formulation are in the form of logic precedence relationship of activities. Besides, the constraints of number of available crew formations which are bound by a lower value and an upper value.

3.2 Solution Approach (MOCS)

The schedule and cost optimisation model of a repetitive construction project aims at selecting the alternative crew formations to minimize both of time and cost. Apparently the optimization model of such projects becomes multi-objective optimization model due to having two objectives to be minimized the same time. The optimization model considers the crew formations used for activities throughout the whole repetitive units as well as interruption days after each activity per unit. This multi-objective optimization ends up finding the trade-off between project duration and direct cost in the form of Pareto optimal curve. Consequently project manager uses the Pareto optimal curve to determine the most suitable optimal solutions that satisfies cost and duration trade-off for the repetitive project.

MOCS is used to find best possible solutions due to its approved efficiency in finding diverse solutions. This algorithm simulates the parasitism life style of the cuckoo birds that lay eggs in host birds. The cuckoo mimics the colours and patterns of the host birds to insure that the host birds will not discover the cuckoo eggs and throw it. The cuckoo birds build cumulative intelligence in selecting the nests. Therefore, algorithm starts at initial solution and start iterations to improve it. Every time, a new solution is created by random Lèvy Flight and then empty nests randomly with a discovery rate. When a new solution is created, it is compared with the original to keep the best and find the Pareto optimum solutions.

3.3 Experimentations and Results

There are two case studies, the first is benchmark project defined as three span concrete bridges and the second is an actual housing project made by the ministry of housing in Bahrain. Both of the case studies aimed at finding the best crew formation and the interruption days that minimizes time and direct cost.

The first case study is consisting of 4 repeating units and the crew formation data and qualities are taken from the originator reference. The Pareto-front is found to compare it with the solution found by (Hyari et. al. 2009). After finding the trade-off solutions, higher level of information are used to find one final results. This information includes indirect costs such as the completion incentives, liquidated damages and lane rental cost. The final solutions after considering each of these cost is compared the one selected by (Hyari et. al. 2009).

In the other hand, the second Case Study consists of four activities and 156 units. It is solved in two parts. In the first (2A), the algorithm is used to find the best crew formations and best interruption days. In the second part (2B), in determine the most suitable contractor for the project in addition to the other decision variables. The result of this part is important to suggest

3.4 Discussion and Conclusion

The results obtained using MOCS Algorithm are compared with those made by (Hyari et. al, 2009) where three span concrete bridge was originally considered. The results compared based on the diversity of solutions and including more dominating solutions. The comparison then includes higher level of information. That is, completion incentives, liquidated damages and lane renal cost. The MOCS solution is considered better that Hyari et. al's solutino when the cost obtained by MOCS is less that the cost obtained by Hyari et. al.

Moreover, the proposed method finds solutions that trade-off time and cost for a housing project in Bahrain which is in turn to be compared with the actual time and cost. Furthermore, the study will also seek possibility of using proposed method in tendering stage to select best contractors who can do the project at the minimum time and costs. This is carried out by extending the second case study to include the decision of who are best constructors.

Chapter 4

Problem Formulation and Modelling

4.1 Parameters

4.2 Decision Variables

4.3 Scheduling Module

4.4 Cost Module

4.5 Modelling of Repetitive Construction Project Schedule and Cost

Chapter 4

Problem Formulation and Modelling

4.1 Parameters

In this study construction project of a concrete bridge with three spans and four repeating segments is considered. Each segment needs five activities to be completed which are excavation, foundation, columns, beams and slabs. These are called repetitive activities because they are repeating over different units. The units in the table are four at which each activity takes place. The quantity of materials ($Q_{i,j}$) in units where i is the activity number and j is the unit number. The value of i is 1 for excavation, 2 for foundation, 3 for columns, 4 for beams and 5 for slabs. The value of j is 1 for the first unit, 2 for the second unit, 3 for the third unit and 4 for the fourth unit. The crew formation (n_i) available is different for each activity. For instance, in excavation there is only one crew formation $n_1 = 1$, while crew formation for foundation n_2 are three crew formations so that the values of n_2 ranges from 1 to 3. Each of these crew formations has certain productivity (${P_i}^n$), labour cost per day (${LC_i}^n$) and equipment cost per day ${LC_i}^n$.

The data used including quantities of work required in the activities, available crew formations in the project with associated productivities and costs are shown in Figure (10). The data in this figure are tabulated to introduce the decision variables. They are called parameters because they are fixed along the programming. Table (1) shows the quantities of the materials in each unit or segment in the project. The data in Tables (2),

(3) and (4) are related with crew formation or options in terms of the productivities, labour costs, and equipment costs respectively.

Table (1)
Quantity of materials in cubic meters

	Unit1	Unit2	Unit3	Unit4	Material Cost
1.Excavation	1147	1434	994	1529	0
2.Foundation	1032	1077	943	898	92
3.Columns	104	86	129	100	479
4.Beams	85	92	101	80	195
5.Slabs	0	138	114	145	186

Table (2)
Productivities of each crew formation in cubic meters per day

	Crew Formation1	Crew Formation2	Crew Formation3	Crew Formation4
1.Excavation	91.75	0.00	0.00	0.00
2.Foundation	89.77	71.81	53.86	0.00
3.Columns	5.73	6.88	8.03	0.00
4.Beams	9.90	8.49	7.07	5.66
5.Slabs	8.73	7.76	0.00	0.00

Table (3)
Labour cost of each crew formation in USD per day

	Crew Formation1	Crew Formation2	Crew Formation3	Crew Formation4
1.Excavation	340	0	0	0
2.Foundation	3804	2853	1902	0
3.Columns	1875	2438	3000	0
4.Beams	3931	3238	2544	1850
5.Slabs	2230	1878	0	0

Table (4)
Equipment cost of each crew formation in USD per day

	Crew Formation1	Crew Formation2	Crew Formation3	Crew Formation4
1.Excavation	566	0	0	0
2.Foundation	874	655	436	0
3.Columns	285	371	456	0
4.Beams	315	259	204	138
5.Slabs	177	149	0	0

4.2 Decision Variables

The decision variables are selected as crew formation/ execution option and interrupted days between each activity in line with (Zhang et. al, 2013). Therefore, number of decision variables used in the decision making problem is I+ ((I-2)x(J-1)) where I are the number of activities and m are the number of units. Therefore, there are 14 variables because I=5 and J=4. The numbers of decision variables for the example taken from (Hyari and El-Rayes, 2006) are listed below. Specifically, the variables are the formation number used in excavation, formation number used in foundations, formation number used in columns, formation number used in beams, and formation number used in slabs, then the rest of variables are for the number of interruption days of activities i after units j. This interruption occurs due to the crew waiting for previous unfinished activity i unit j. the values of i is ranging from i=1 for the first unit maximum value I=5 and the value of j ranges from i=1 at first unit until J= 4 for the last unit. Therefore, there are fourteen integer variables, 5 of them are attributes of representing the crew formation number. The other variables i.e. from 6 to 14 are interruption time in days. These decision variables are bounded by lower bounds and upper bounds which

are non-negative. These are considered as non-negative variable inequality constraint.

The definition of variable are listed as follows:

- x_i: represents crew formation in activity i
- $inter_{i,j}$: represents interruption days in activity i after unit j

The two objective functions to be considered are the time and cost. The time function is related to the quantities of work and the productivities of the crew formations which determines the duration of the activities. The cost function calculates the direct costs of the projects which depend on the cost of the resources per day, productivity of the resources, the quantity of materials and the cost rate of materials. From these data, the direct costs are calculated such as material cost in activity i unit j $MC_{i,j}$, labour cost in activity i unit j $LC_{i,j}$, and equipment cost in activity i unit j $EC_{i,j}$. The time function is mainly called scheduling module because it finds the schedule needed to finish all the activities throughout all units. It worth pointing out that this formulation is used for units arranged to be built in series.

4.3 Scheduling Module

Durations are determined based on the productivity of crew formations and the quantities of work in each repeated activity throughout a construction project. The scheduling model is made to generate a resource-driven schedule that complies with logical relationship, crew

availability and crew work continuity constraints. A sample of calculations is shown in Figure (11).

Within this study, the approach of (Hyari and El-Rayes, 2006) is adopted for scheduling calculations. The approach is based on seven steps:

1. Duration ($d_{i,j}$) of each activity (i) per unit (j) is measured according to the equation below from the productivity in units per hours and on the quantity of materials ($Q_{i,j}$) in units.

$$d_{i,j} = \frac{Q_{i,j}}{P_i^n} \; \forall \; i \; and \; j \qquad \text{Equation 5}$$

The productivity depends on the decision variables x1-x5 which makes it important to use a factor that substitutes the right value of the productivity based on the value of $x1$- $x5$. In excavation there is only one method available, so it is directly divided by one productivity value which is 91.75cubic meter per day. In the other activities, however, the productivities are varying from 2 to 4 options of crew formations. For example, in foundations, the term $[89.77\,[(2-x2)(3-x2)]/2 + 71.81[(x2-1)(3-x2)] + 53.86\left[\frac{(x2-1)(x2-2)}{2}\right]]$ is a factor used to output the related productivity P_2^{x2}. Therefore, when the value of x2 is substituted as 1 it gives the productivity equal to 89.77 cubic meter per day, when it is equal to 2, productivity becomes 71.81 cubic meters per day and when it is 3 the productivity becomes 53.86 cubic meters per day. The mathematical expressions of durations for unit 1 and 2 are as follow:

Excavation:

$$d_{1,1} = \frac{1147}{91.75\,x1}$$

$$d_{1,2} = \frac{1434}{91.75\ x1}$$

Foundation:

$$d_{2,2} = \frac{1032}{89.77[(2-x2)(3-x2)]/2+71.81[(x2-1)(3-x2)]+53.86\left[\frac{(x2-1)(x2-2)}{2}\right]}$$

$$d_{2,2} = \frac{1077}{89.77[(2-x2)(3-x2)]/2+71.81[(x2-1)(3-x2)]+53.86\left[\frac{(x2-1)(x2-2)}{2}\right]}$$

Column:

$$d_{3,1} = \frac{104}{5.73[(2-x3)(3-x3)]/2+6.88[(x3-1)(3-x3)]+8.03\left[\frac{(x3-1)(x3-2)}{2}\right]}$$

$$d_{3,2} = \frac{86}{5.73[(2-x3)(3-x3)]/2+6.88[(x3-1)(3-x3)]+8.03\left[\frac{(x3-1)(x3-2)}{2}\right]}$$

Beams:

$$d_{4,1} = \frac{85}{9.9[(2-x4)(3-x4)(4-x4]/6+8.49[(x4-1)(3-x4)(4-x4)/2]+7.07[(x4-1)(x4-2)/2]+5.66[(x4-1)(x4-2)(x4-3)]}$$

$$d_{4,2} = \frac{92}{9.9[(2-x4)(3-x4)(4-x4]/6+8.49[(x4-1)(3-x4)(4-x4)/2]+7.07[(x4-1)(x4-2)/2]+5.66[(x4-1)(x4-2)(x4-3)]}$$

Slabs:

$$d_{5,1} = \frac{0}{8.73(2-x5)+7.76(x5-1)}$$

$$d_{5,2} = \frac{138}{8.73(2-x5)+7.76(x5-1)}$$

2. Calculation of times of earliest start $S_{crew(i,j)}$ and finish $F_{crew(i,j)}$ for activity (i) in unit (j) that to satisfy crew availability and work continuity constraints are in the relationships shown in the equations 6 to 8. The completion $F_{crew(ij)}$ of an activity (i) in a unit (j) is after the duration $d_{i,j}$ is passed after the start time $S_{crew(ij)}$ of this activity. In the same time, that start time of an activity in any unit $S_{crew(ij)}$ will be after the completion $F_{crew(i(j-1))}$ of the previous unit (j−1) together with the nominated interruption time$inter_{(i(j-1))}$. Due to the fact the first unit has

no predecessors in the first units (i, 1) of each activity, the start time is zero.

$$F_{crew(ij)} = S_{crew(i,j)} + d_{i,j} \quad \forall\, i \text{ and } j \qquad \text{Equation 6}$$

$$S_{crew(i,1)} = 0 \quad \forall\, i \text{ and } j \qquad \text{Equation 7}$$

$$S_{crew(i,j)} = F_{crew(i,(j-1))} + inter_{(i,(j-1))} \quad \forall\, i \text{ and } j \qquad \text{Equation 8}$$

Sample of calculations for first activity and first unit is as follows:

$$F_{crew(1,1)} = S_{crew(1,1)} + d_{1,1}$$

$$S_{crew(1,2)} = F_{crew(1,1)} + inter_{(1,1)}$$

$$S_{crew(1,1)} = 0$$

3. The third relationship shown in Equation 9 is to satisfy the job logic or precedence relationships between any activity (i) and all its predecessors activates like finish to start, start to start, finish to finish or start to finish with or without lag time. The constraint shown in Equation 9 is a finish to start constraint, i.e. the successor activity (i) in each unit (j) can start only after the completion of its predecessor activity (i-1). This constraint is initiated to be equal and then updated.

$$S_{logic(ij)} \geq F_{((i-1),j)} + lag_{((i-1),j)} \quad \forall\, i \text{ and } j \qquad \text{Equation 9}$$

Sample of calculations for first and fifth activities, respectively, at first unit are as follow:

$$S_{logic(2,1)} \geq F_{(1,1)} + lag_{(1,1)}$$

$$S_{logic(5,1)} \geq F_{(4,1)} + lag_{(4,1)}$$

4. In this step, the difference between the values obtained in the last two steps calculated as shown in the relationship in Equation 10 which aims to maintain the required delay in $S_{crew(ij)}$ and $F_{crew(ij)}$ to further satisfy the job logic constraint.

$$\Delta_{ij} = S_{logic(ij)} - S_{crew(i,j)} \quad \forall\, i \text{ and } j \qquad \text{Equation 10}$$

Sample of calculations for first and first unit is as follows:

$$\Delta_{(1,1)} = S_{logic(1,1)} - S_{crew(1,1)}$$

5. In this step one value of the difference will be considered to shift an activity (i) along the whole units (j) to identify the scheduled start $S_{(ij)}$ and finish $F_{(ij)}$times to satisfy all scheduling constraints.

$$Shift_i = Max_{j=1}^{J} [\Delta_{ij}] \quad \forall\, i \text{ and } j \qquad \text{Equation 11}$$

For example, the shift in the first activity is as follows:

$$Shift_1 = Max\, [\Delta_{1,1}, \Delta_{1,2}, \Delta_{1,3}, \Delta_{1,4}]$$

Then, the start time and finish time of the activities are calculated as follows:

$$S_{(ij)} = S_{crew(ij)} + Shift_i \qquad \forall\, i \text{ and } j \qquad \text{Equation 12}$$

$$F_{(ij)} = F_{crew(ij)} + Shift_i \qquad \forall\, i \text{ and } j \qquad \text{Equation 13}$$

$S_{i,j}$= start time of activity (i) in repetitive unit (j);

$F_{i,j}$= finish time of activity (i) in repetitive unit (j);

For example for first activity and first unit:

$$S_{(1,1)} = S_{crew(1,1)} + Shift_1$$

$$F_{(1,1)} = F_{crew(1,1)} + Shift_1$$

6. This step is just to find the duration (D) of the schedule obtained in the previous steps by just subtracting the completion time of the last activity (I) in the last unit (J) from the start of the first activity in the first unit $S_{(1,1)}$.

$$D = F_{(I,J)} - S_{(1,1)} \qquad \text{Equation 14}$$

D= project total duration;

This step obtains the first objective function ,project duration, which is the difference between the finish time of the last activity last unit and

the starting time of the first activity, first unit. This is applied in the problem as follows:

$D=F_{5,4} - S_{1,1}$

7. The interruption is calculated in two steps is the and basically use the values of interruptions nominated $inter_{ij}$ occur after each activity (i) per unit (j). The first step is to find interruption $inter_i$ for each activity (i) which is simply the summation of the interruptions along the units in that activity and the second calculation is the total time interruptions R in the entire project.

$$inter_i = \sum_{j=1}^{J} inter_{i,j} \quad \forall\, i \text{ and } j \qquad \text{Equation 15}$$

$$R = \sum_{i=1}^{I} inter_i \quad \forall\, i \text{ and } j \qquad \text{Equation 16}$$

DC =project direct cost;

An example of the interruptions in the first activity is as follows:

$$inter_1 = inter_{1,1} + inter_{1,2} + inter_{1,3} + inter_{1,4}$$

The application of the R in the project:

$$R = inter_1 + inter_2 + inter_3 + inter_4 + inter_5$$

4.4Direct Cost Module

Based on the decision variables, the direct costs are calculated to be included later in the objective function. These costs are dependent on the amount of work or amount of material that are processed in a certain productivity rate. Thus, the cost rates are selected based on the activity (i) productivity (P_i^n) alternative (n) and are multiplied by the duration by which an activity will be completed. Therefore, the duration of an activity (i) per unit (j) is the key value that represents the time needed to accomplish it and is calculated in equation 5. The project direct cost includes activity dependent costs throughout repetitive units in

construction project, which are material costs, labour costs and equipment costs. The total cost of the whole project is defined as follows:

$$DC = \sum_{i=1}^{I}\sum_{j=1}^{J}(MC_{i,j} + LC_{i,j} + EC_{i,j}) + IRC \qquad \text{Equation 17}$$

$MC_{i,j}$ =Material cost for performing activity (i) in repetitive unit (j);

$LC_{i,j}$ =Material cost for performing activity (i) in repetitive unit (j);

$EC_{i,j}$ =Equipment cost for performing activity (i) in repetitive unit (j);

IRC =Interruption cost;

4.4.1 Material Cost

The material cost of activity i unit j is calculated based on the qualities of material and the cost rate of that material which are mathematically defined as follows:

$$MC_{ij,} = Q_{ij,}M_i \quad \forall\, i \; and \; j \qquad \text{Equation 18}$$

Where

M_i =Material cost per cubic meter in activity (i);

An example of the second activity and first unit is as follows:

$$MC_{2,1} = (132) \times 92$$

The total costs of materials are calculated as follows:

MC=(132+1077+943+898) ×92+(104+86+129+100) ×479+(85+92+101 +80) ×195+(138+114+145) ×186= **$ 624,953**

4.4.2 Labour Cost

The labour cost ($LC_{i,j}$) is calculated based on duration ($D_{i,j}$) the labours will spend to accomplish the activity (i) in each unit (j) and on the labour cost rate (C_i^n) based on the option (n) which are mathematically defined as below:

$$LC_{ij,} = d_{i,j}C_i^n \qquad \forall\, i \text{ and } j$$ Equation 19

Where

C_i^n= Labour cost per day for performing activity (i) using method n;

The term $LC_{(Activity\ i,\ Unit\ j)}$ is used to represent the labour cost for each activity per each unit and is related to labour cost rate. Apparently, labour costs rate is also dependant on the decision variables $x1$- $x5$ which are the identification numbers of the crew formation selected. Therefore, the factors are also needed to substitute the value of the decision variables x1-x5 to find the cost rate. For example in foundations, the term $[3804\,[(2-x2)(3-x2)]/2 + 2853[(x2-1)(3-x2)] + 1902\left[\frac{(x2-1)(x2-2)}{2}\right]]$ is the factor C_2^{x2} used to find the correct labour cost rate. That is, when the decision value of x2=1, the cost rate is \$3804/day, when $x2$=2, the cost rate is \$2853/day and when x2=3, the cost rate is \$1902/day. An example is for second activity first unit as follows:

$$\mathrm{LC}_{2,1} = d_{2,1} \times [3804\,[(2-x2)(3-x2)]/2 + 2853[(x2-1)(3-x2)] + 1902\left[\frac{(x2-1)(\mathrm{x2}-2)}{2}\right]]$$

4.4.3 Equipment Cost

Equipment cost ($EC_{i,j}$ is the cost incurred due to use any machinery, tools, and apparatus. The cost is generally calculated based on the duration ($d_{i,j}$) needed to accomplish an activity (i) per unit (j), and on the cost rate E_i^n that refer to the option (n) of potential costs incurred such as rental cost, fuel, purchase, or maintenance, .etc. the relationship is mathematically defined below

$$EC_{i,j} = d_{i,j}E_i^n \qquad \forall\, i \text{ and } j$$ Equation 20

Where

E_i^n= Equipment cost per day for performing activity (i) using method n;

The term $EC_{(Activity\ i,\ Unit\ j)}$ is used to represent the equipment cost for each activity per each unit and is related to equipment cost rate. Apparently, equipment costs rate is also dependant on the decision variables x1-x5 which are the identification numbers of the crew formation selected. Therefore, the factors are also needed to let substitute the value of the decision variables x1-x5 to find the cost rate. For example in foundations, the term $[874[(2-x2)(3-x2)]/2 + 655[(x2-1)(3-x2)] + 436\left[\frac{(x2-1)(x2-2)}{2}\right]]$ is the factor E_2^{x2} used to distinguish between the crew formations. That is when the first crew formation corresponds to cost rate of $ 874/day, the second for $655 / day and the third is corresponds to $436/day. An example for second activity first unit is as follows:

$EC_{2,1}=d_{2,1}\times [874[(2-x2)(3-x2)]/2 + 655[(x2-1)(3-x2)] + 436\left[\frac{(x2-1)(x2-2)}{2}\right]]$

4.4.4 Interruption Cost

The interruption cost IRC is the cost that is incurred due to the crew being idle or unproductive and is related to the interruption for each activity $inter_i$ which is calculated in the last step in the scheduling model.

$$IRC = inter_i . (C_i^n + E_i^n) \qquad \text{Equation 21}$$

IRC= cost due to resources being idle during schedule interruptions;

$inter_i$= interruption occurs in activity i;

This is simply interpreted as following, bearing in mind that the brackets are including the same factors explained in the labour and equipment costs.

$$IRC = inter_1 . (C_1^{x1} + E_1^{x1}) + inter_2 . (C_2^{x2} + E_2^{x2}) + inter_3 . (C_3^{x3} + E_3^{x3}) + inter_4 . (C_4^{x4} + E_4^{x4}) + inter_5 . (C_5^{x5} + E_5^{x5})$$

4.5 Modelling of Repetitive Construction Project Schedule and Cost

Decision Variables

x_i: represents crew formation in activity i

$inter_{i,j}$: represents interruption days in activity i after unit j

$where\ i = 2,3,4\ ; j = 1,2,3$

Objective Functions

When applying the equations on the problem described, the following formulation is used:

$$\text{Min } f_1 = F_{5,4} - S_{1,1} \qquad \text{Equation 22}$$

$$\text{Min } f_2 = \sum_{i=1}^{5} \sum_{j=1}^{4} (MC_{i,j} + LC_{i,j} + EC_{i,j}) + IRC \qquad \text{Equation 23}$$

Where $MC_{i,j}$, $LC_{i,j}$, $EC_{i,j}$, IRC, $F_{5,4}$, and $S_{1,1}$ are obtained from the equality and inequality constraints as will be explained in the following parts:

Subjected to:

Parameters:

Total Material Cost (sum of all material costs):

MC=(132+1077+943+898) ×92+(104+86+129+100) ×479+(85+92+101 +80) ×195+(138+114+145) ×186= **$ 624,953**

$P_1^{x1} = 91.75\ x1$

$$P_2^{x2} = 89.77\,[(2-x2)(3-x2)]/2 + 71.81[(x2-1)(3-x2)] + 53.86\left[\frac{(x2-1)(x2-2)}{2}\right]$$

$$P_3^{x3} = 5.73\,[(2-x3)(3-x3)]/2 + 6.88[(x3-1)(3-x3)] + 8.03\left[\frac{(x3-1)(x3-2)}{2}\right]$$

$$P_4^{x4} = 9.9\,[(2-x4)(3-x4)(4-x4]/6 + 8.49\left[\frac{(x4-1)(3-x4)(4-x4)}{2}\right] + 7.07[(x4-1)(x4-2)/2] + 5.66[(x4-1)(x4-2)(x4-3)]$$

$$P_5^{x5} = 8.73(2-x5) + 7.76(x5-1)$$

$$C_1 = 340\ x1$$

$$C_2 = [3804\,[(2-x2)(3-x2)]/2 + 2853[(x2-1)(3-x2)] + 1902\left[\frac{(x2-1)(x2-2)}{2}\right]]$$

$$C_3 = [1875\,[(2-x3)(3-x3)]/2 + 2438[(x3-1)(3-x3)] + 3000\left[\frac{(x3-1)(x3-2)}{2}\right]]$$

$$C_4 = [3931\,[(2-x4)(3-x4)(4-x4]/6 + 3238\left[\frac{(x4-1)(3-x4)(4-x4)}{2}\right] + 2544[(x4-1)(x4-2)/2] + 1850[(x4-1)(x4-2)(x4-3)]]$$

$$C_5 = [2230(2-x5) + 1878(x5-1)]$$

$$E_1 = 566\ x1$$

$$E_2 = [874\,[(2-x2)(3-x2)]/2 + 655[(x2-1)(3-x2)]$$

$$+436\left[\frac{(x2-1)(x2-2)}{2}\right]]$$

$$E_3 = [285\,[(2-x3)(3-x3)]/2 + 371[(x3-1)(3-x3)]$$

$$+456\left[\frac{(x3-1)(x3-2)}{2}\right]]$$

$$E_4 = [315\,[(2-x4)(3-x4)(4-x4]/6$$

$$+259\left[\frac{(x4-1)(3-x4)(4-x4)}{2}\right]$$

$$+204[(x4-1)(x4-2)/2]$$

$$+148[(x4-1)(x4-2)(x4-3)]]$$

$$E_5 = [177(2-x5) + 149(x5-1)]$$

$$d_{i,j} = \frac{Q_{i,j}}{P_i^n} \qquad \forall \text{ i and j}$$

$LC_{i,j}=d_{i,j}\times C_i \quad \forall$ i and j

$EC_{i,j}=d_{i,j}\times E_i \quad \forall$ i and j

Equality constraints

$F_{crew(i,j)} = S_{crew(i,j)} + d_{i,j} \qquad \forall$ i and j

$S_{crew(i,1)} = 0 \qquad \forall$ i

$S_{crew(i,j)} = F_{crew(i,j)} + inter_{(i,j)} \quad \forall$ i and j

$\Delta_{i,j} = S_{logic(i,j)} - S_{crew(i,j)} \quad \forall$ i and j

$Shift_i = Max\,[\Delta_{(i,j)},\Delta_{(i,j)},\Delta_{(i,j)},\Delta_{(i,j)}] \quad \forall$ i and j

$S_{(i,j)} = S_{crew(i,j)} + Shift_i \quad \forall$ i and j

$F_{(i,j)} = F_{crew(i,j)} + Shift_i \quad \forall$ i and j

$inter_i = inter_{i,j} + inter_{i,j} + inter_{i,j} \quad \forall$ i and j

Where i = 2,3,4 ; j = 1,2,3

Inequality constraint:

$S_{logic(i,j)} \geq F_{(i-1,j)} + lag_{(i-2,j)}$

$1 \leq x_1 \leq 1$

$1 \leq x_2 \leq 3$

$1 \leq x_3 \leq 3$

$1 \leq x_4 \leq 4$

$1 \leq x_5 \leq 2$

Non-negativity:

$inter_{i,j} \geq 0 \quad \forall$ i and j

$x_i \geq 0$

where i = 2,3,4 ; j = 1,2,3

** x_1 , $x_{2,..} x_5$ and $inter_{i,j}$ (i = 2,3,4 ; j = 1,2,3) are integers

Chapter 5

Solution Approach (MOCS)

5.1 Optimisation

5.2 Multi-Objective Cuckoo Search Algorithm (MOCS)

Chapter 5

Solution Approach (MOCS)

5.1 Optimization

Optimization is –as explained in Chapter2- the process of searching feasible values of decision variables that produces extreme values in the objective function. It is described as "treasure hunting" (Yang, 2010). The optimization problem may be formulated such that it seeks to maximize or to minimize one objective function while satisfying a number of constraints. This is expressed mathematically in equations (1) to (4) in Chapter 2 where i=1 for single objective function.

5.1.1 Multi-objective Optimization (MOOP)

Multi-objective optimization considers several objective functions simultaneously which are most of the time conflicting in nature and need trade-off. The multi-objective optimization does not present a unique solution, but a set of compromised solutions called Pareto optimal solutions as shown in Figure (15). As illustrated in equations 1 to equations 4 in chapter 2, solutions x trade-off the competing objectives F_i while satisfying the inequalities$h_j(x)$, and equalities$g_k(x)$.

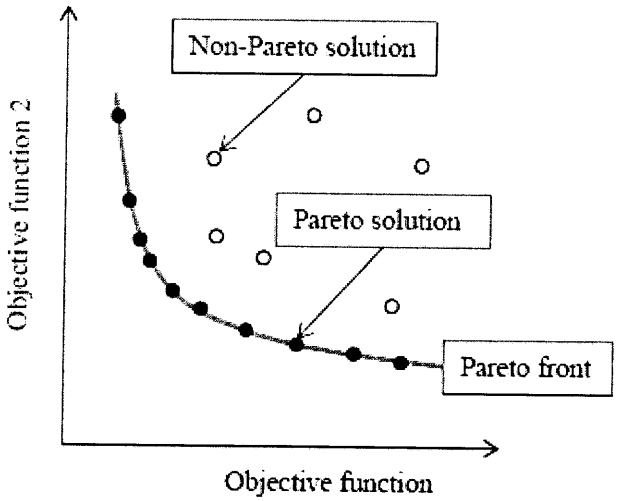


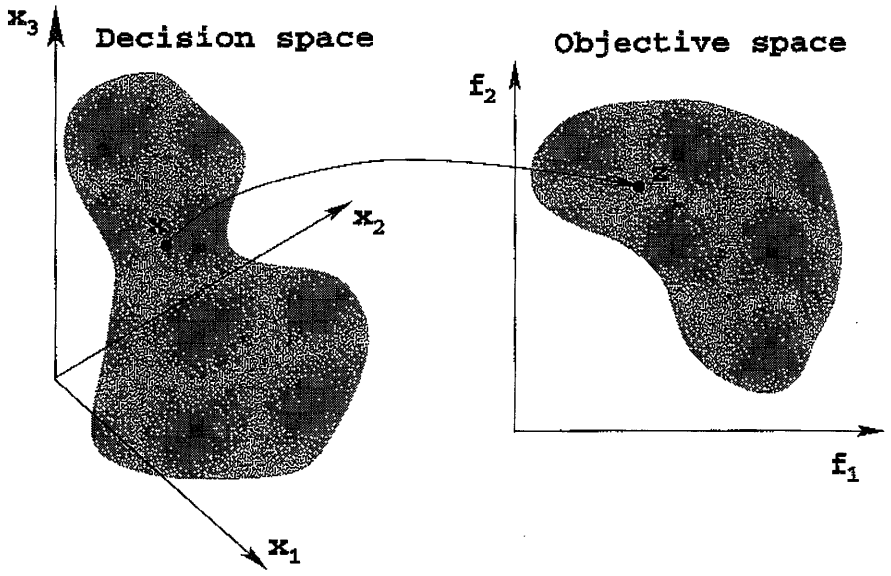
Figure (15)
Pareto solutions

It is not so easy to find an optimum solution from three objectives and impossible in cases with more than three objectives, so the MOOP in this part is concerned with two objectives. These two objective functions are corresponding to several decision variables from decision variables space as shown in Figure (16). This graph (Figure (16) (b)) has the x-axis as the first objective and the y-axis as the second objective. It results in several optimum solutions allowing the decision maker to select one of them. (Deb, 2001). In engineering design problems, functions are expensive to evaluate, as they need several kinds of analyses. The solutions obtained in the engineering designs are obtained by computer and by man in a way that could expose to consistency (Nakayama, 2005). Therefore, the decision maker will prefer an objective and scarify another one based on the additional information called higher level information as shown in Figure (17) (Deb, 2001). Also, it is important in MOOP to generate diverse Pareto-optimal points in the Pareto-front to cover more alternatives in the decision making process (Deb, 2001).

Generally, algorithms used for single objective optimization need modifications to let it work for multi-objective problems such as combining the objectives into one objective using weighted sum. However, the weighted sum faces a problem of generating diverse solutions which does not help to solve real-world at a degree of uncertainty or noise (Yang and Deb, 2011). The diversity in MOOP could come from several techniques that makes good approximation and diversity to Pareto front such as directed search domain explained by (Erfaniaet. al, 2010), genetic labelling algorithms for (Figueira, et. al, 2009), and improved strategies by (Gujarathi1et. al, 2009) .

Otherwise, the application of Lévy flights in this thesis as explained in part 4.3 is made to ensure the good diversity of the solutions in the Pareto-front. (Yang and Deb, 2011)

In the next subparts, the important topic is to find the dominating solutions in order to create the Pareto-optimum using fitness assignments dragged from the logic explained.



(a) (b)

Figure (16)
Illustration of the decision variable space and the corresponding objective space

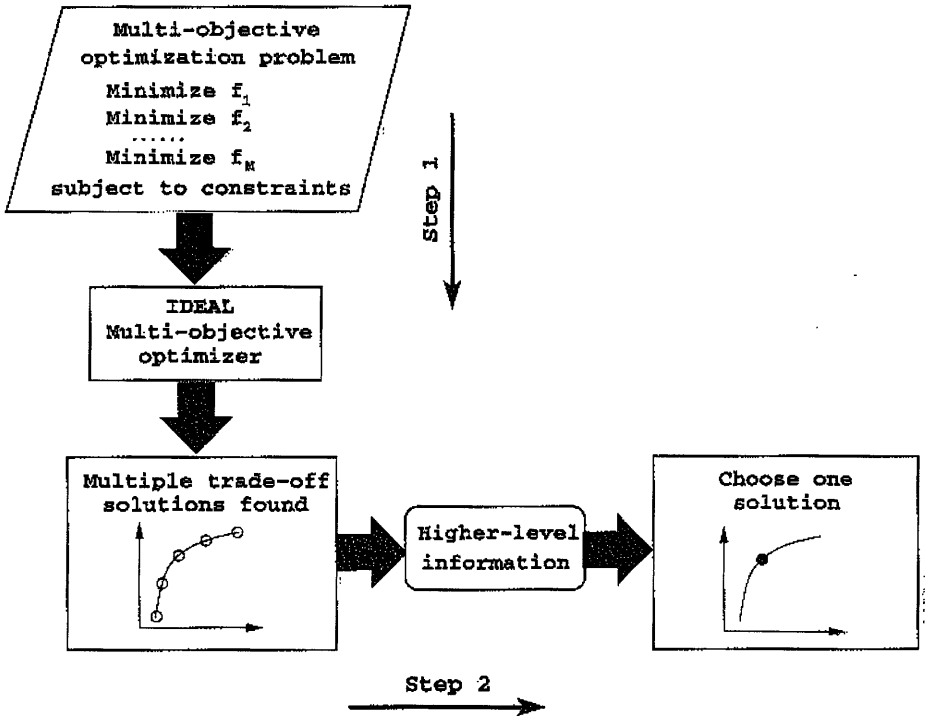


Figure (17)
Steps needed to find one final solution in MOOP

Domination of a Set

The domination is an important logical process that finds the optimum solutions. Domination is decided based on two conditions stated in (Deb, 2001). That is, a solution x1 is said to dominate the other solution x2 if both of the following conditions are true:

1. The solution x1 is no worse than x2 in all objectives
2. The solution x1 is strictly better than x2 in at least one objective

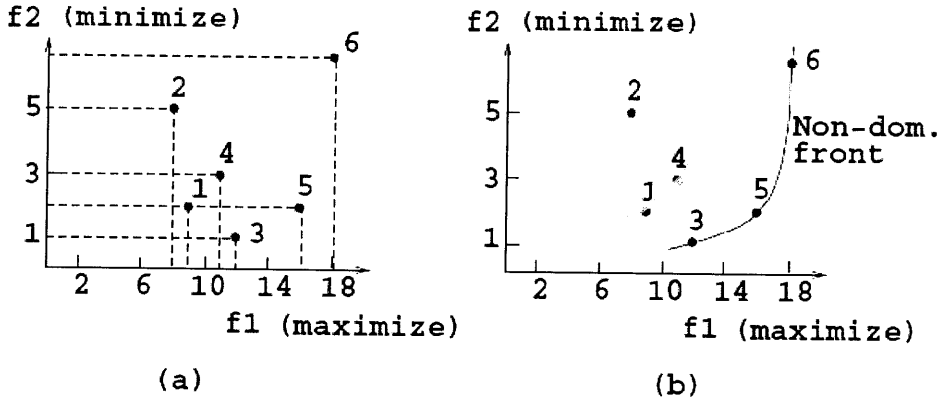


Figure (18)

Population of 6 solutions and its non-dominated set (Deb, 2004)

The dominance concept is defined by (Deb, 2001) and illustrated in an example as shown in Figure (18). This example contains 6 potential solutions for two-objective optimization problem that aims to maximize f1 and minimize f2. When following the dash lines, it is noticed that solution 1 is dominating solution 2 because it produces higher value of f1 than that of solution 2 and a lower value of f2 than that of solution 2. Likewise, it is clear in the figure those solutions 3 and 5 dominating solutions 1, 2, and 4. Furthermore, when comparing solutions 3 and 5 and 6 to each other, there is only one objective that offers better value for

instance, solution 3 produces better value for the second objective function, solution 5 produces better value for the first objective function and solution 6 is better than 3 and 5 in the first objective only. Thus, these solutions are not dominating each other and called non-dominated solutions. To conclude, there are three levels in domination in this graph, amongst which only level1 is considered to be the Pareto- Optimum set. The points belonging to the level are classes as follows:

- Level 1: Points 3, 5, and 6
- Level 2: Points 1 and 4
- Level 3: Point 2

Pareto Front

The single objective optimization aim to find one best solution while the multi-objective optimization problem aims to find a number of diverse optimal solutions presented as a Pareto-optimal front. There are two goals to be achieved to find quality of Pareto-front, 1.convergence and 2.diversity as represented in Figure (19) and Figure (20). In Figure (21), the graphs show the grey area representing the search space and the Pareto-optimal set is the continuous curves which are always in the edges of the solution space. There are several purposes of optimization classified by the combinations of objectives whether minimized or maximized. When the Pareto curve tends to minimize both objectives then it will move towards the boundary in the top-left graph. When f1 is the minimized objective while maximizing f2, however, the graph will look like the one in the top-right. Furthermore, the Pareto-optimal sets for maximizing f1, minimizing f2 and maximizing f1, maximizing f2 are also

displayed in the bottom-left and bottom-right figures, respectively. (Deb, 2001)

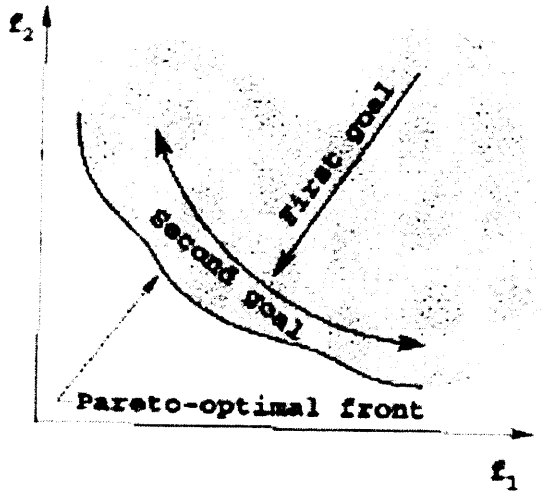


Figure (19)
Two goals in MOOP

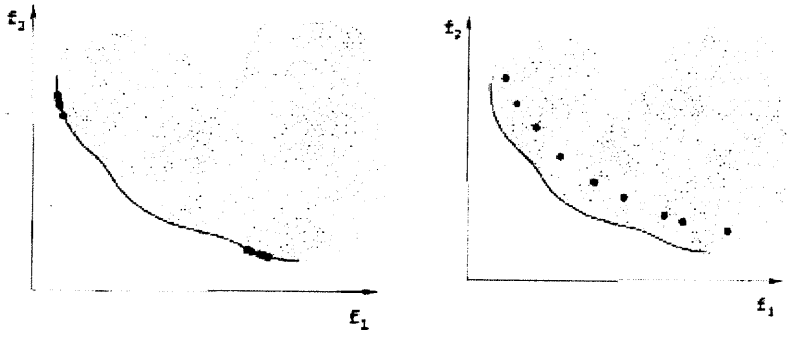


Figure (20)
Distribution and Conversion: The left figure shows poor distribution and the other figure shows poor convergence (Deb, 2001)

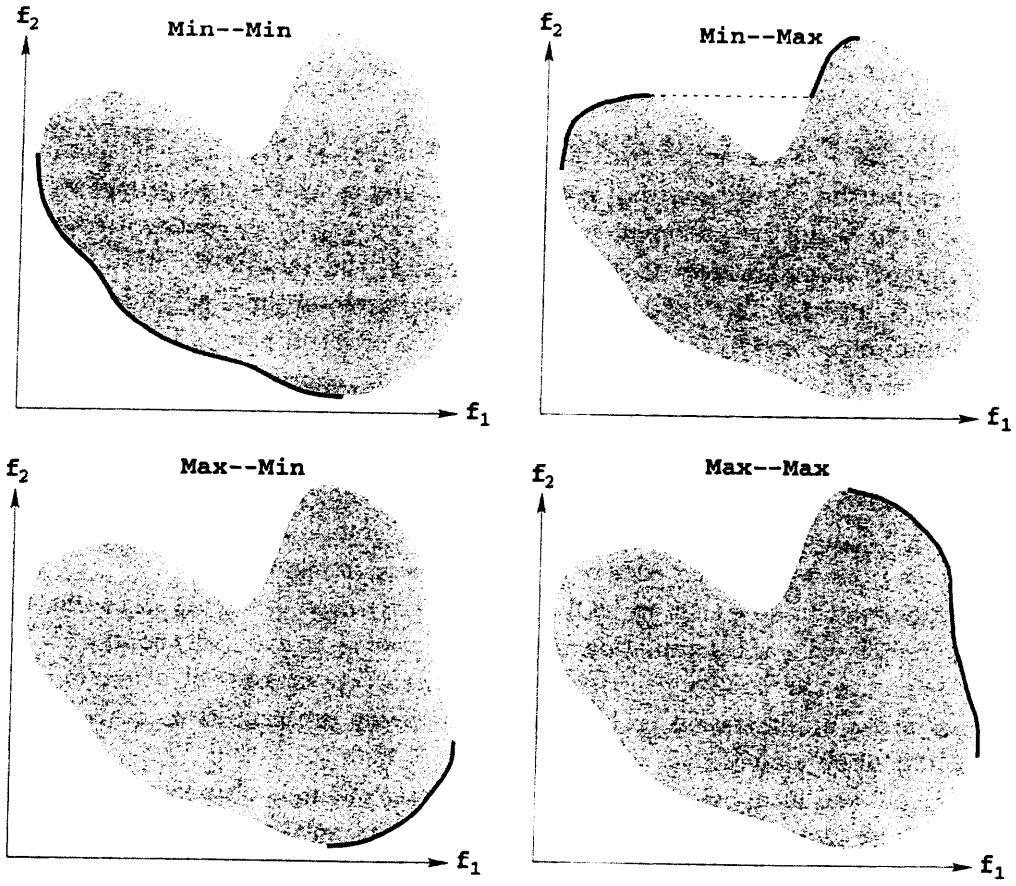


Figure (21)
Pareto-optimal for four arrangements of two types of objectives (Deb, 2001)

When the optimization targets minimal values for both objectives, then the Pareto solutions will target minimum values for both objective within solutions. The comparison will be carried on all solutions along all objectives. For instance, if we take two solutions to compare with each other such as vector $\boldsymbol{u}$ and another vector $\boldsymbol{v}$ and both vectors contain all the objective functions. Then the following condition will let decide upon which one is dominating the other:

Solution vector $\boldsymbol{u}$ is called dominating if and only if $u_i \leq v_i$ for $\forall\, i \in \{1, \ldots n\}$ and $\exists\, i \in \{1, \ldots n\} : u_i < v_i$. In other words, no

component of $\boldsymbol{u}$ is larger/ worse than the corresponding component of $\boldsymbol{v}$, and at least one component is smaller. This process aims to find the non-dominated set which is not dominated by any other member. This set is called globally Pareto-optimal set when it is the non-dominated by the entire feasible search. Otherwise solutions could be locally Pareto-optimal set as shown in Figure (22). This case is tested when every x in a set has no neighbourhood y with dominating members, such that the absolute difference bet*ween* x and y is a small positive number ε.

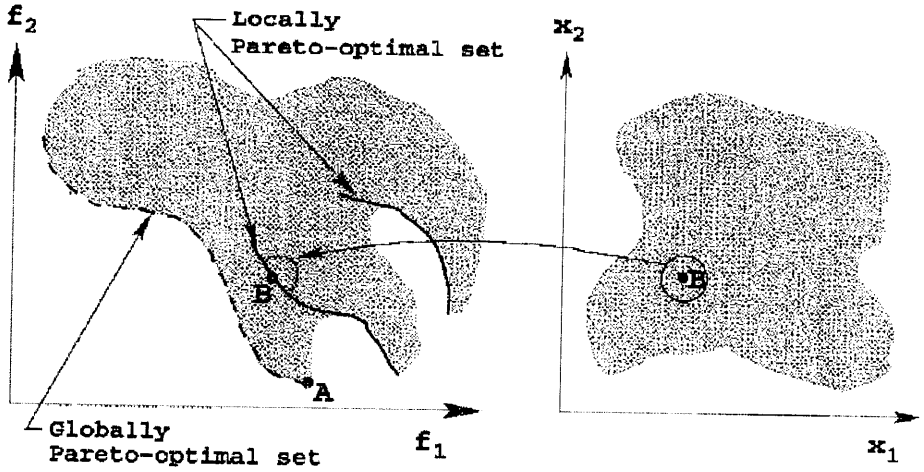


Figure (22)

Illustration of local such as solution (B) and global Pareto-optimal solutions (Deb, 2001)

5.2 Optimization Using Cuckoo Search Algorithm

CSA mimics the interesting breed behaviour of certain cuckoo species. Cuckoos are fascinating birds, not only because of the beautiful sounds they can make, but also because of their aggressive reproduction strategy. Some species of cuckoo birds lay their eggs in communal nests. In some cases they may even remove others' eggs to increase the

hatching probability of their own eggs. CSA imitates this breeding behaviour as a numerical optimization technique.

5.2.1 Cuckoo Birds Lifestyle

The cuckoo birds are with a dove size bird with blue grey upper parts and dark barred white under parts. Although the cuckoos mostly occur in forests and woods like the rainforests of the tropics, but there are some which occupy arid areas like deserts. The cuckoo best environment is the one provide food –insects- and a breeding place which is the environment of parasitized host species. Generally, this parasitism of the cuckoos can take place in three different forms depending on the specie:

1. Intraspecific brood parasitism
2. Cooperative breeding
3. Nest takeover



Figure (23)
Cuckoo Bird from (BBC Wildlife Video) (Yang, 2010)

The common cuckoos are also migrate birds in some seasons like the one migrate to the UK in the summer. They are not only special with their sound, "sleek body, long tail and pointed wing" but also with their intelligent parasitism life style. Naturally, the cuckoo birds never build their own nests but manage to lay their eggs in the nest of another bird species like meadow pipits, dun nocks and reed warblers. These host birds will care for their chicks until they grow up rapidly and return back home. There are several tricks the cuckoo performs to parasitize in other birds species nests:

1. Some cuckoo birds can parasitize the nests of different types of bird species to mimic the colour and pattern of the host birds' eggs to lay their eggs so the host birds keep it as its own eggs.
2. Another way is that female cuckoos specialize only on a particular host species and manage to parasitize in its nest. This case is when the cuckoos are grouped into gents; each is specialized in a particular host that the cuckoos imitate their eggs. Moreover, there are cuckoo species such as Ani and Guira which lay eggs in communal nests after they remove others' eggs to reduce the probability of discovering their eggs. One of the natural mysteries is how the cuckoo imitates the host's eggs so accurately so that the host bird never notices its eggs.
3. Moreover, the cuckoo egg hatches earlier than the host's, and grows even faster; which makes them more demanding to the extent that sometimes they instinctively evict the hosts' eggs out to get the furthermost care from the host birds.

4. When more eggs hatch and chicks grow up in an area, the cuckoo birds will perceive this as an indication for the appropriateness of the nests in that area.

In the other hand, the host species that experienced the cuckoo parasitism in their nests performs preventions that could reach up to mobbing the cuckoos and drive them out of the area. Otherwise, the birds need to know how to find the cuckoo egg correctly and respond to that before it is too late. Many host birds successfully could identify a cuckoo egg that is laid in their own nest so they either when a bird is practically suffering an egg in their nest they react to it in two different ways:

1. When they detect the cuckoo eggs they throw it outside the nest to kill it.
2. Demolish the whole nest and rebuild it in another place.

Nevertheless, what is noticed over time is that the cuckoo constantly improves its mimicry of its hosts' eggs, and the hosts attempt to discover means to distinguish the parasitic egg. Also, they search for the most proper area to lay their eggs to maximize their survival possibility. They immigrate toward best habitats as they will live somewhere near to it. That is, when a cuckoo chick grows to a mature cuckoo, they make societies with their own habitat area to live in and naturally will be the target for other societies. Therefore, when cuckoo starts laying eggs randomly and considering the number of eggs and the laying radius until the best position with most successful hatching and grown cuckoos is gathered around (Rajabioun, 2011); (Anon., 2014); (Yang and Deb, 2011).

5.2.2 Single Objective Cuckoo Search Algorithm (CSA)

The algorithm mimics the intelligence in the breeding of cuckoo birds in their randomized laying of eggs in foreign nests where there is a potential danger for the eggs to be discovered and be expelled.

In CSA, a cuckoo egg represents a potential solution to the design problem which has a fitness value.

The algorithm follows three guidelines:

a) Each cuckoo lays one egg at a time in a random nest.
b) The best nests with high quality eggs will carry over to the next iterations.
c) The number of available host nests is fixed, and a host can discover an alien egg with a probability Pa, a random number within [0, 1]. In this case, the host bird can either throw the egg away or abandon the nest so as to build a completely new nest in a new location.

CSA requires a matrix consists of the population of 'n' nests representing the potential solution(s) to the problem and 'm' decision variables per solution. Every solution has its objective value(s) which is representing k number of eggs. The algorithm then generates a new solution based on the previous and a random walk step. It worth pointing out that the selection of the best nest by keeping its solution for next generations (exploitation) corresponds to a form of elitism in the Gas (Yang and Deb, 2010). This step aims to generate new exploiting move using Lévy flights alone or together with normal distribution and

Gaussian used in other algorithms. The Lévy flight is considered as one of the stochastic process of traveling objects along random trajectories like the search route of a hunting animal. For the reason Lévy flight calculates the step lengths based on a particular probability distribution, and the directions of the steps. (Saka et. al, 2013);(Yang and Deb, 2010)

The following equations are used to calculate the new solutions generated after calculating new steps:

$$x_i^{(t+1)} = x_i^{(t)} + step \qquad \text{Equation 24}$$

$$\text{step} = \alpha \otimes Lévy(\beta) \qquad \text{Equation 25}$$

Where β is the step size parameter and $\alpha>0$ is the step size related to the scales of the problem which in most cases is $\alpha = O(1)$ and is calculated as follows:

$$\alpha = \alpha_o(x_i - x_j) \qquad \text{Equation 26}$$

The product $\otimes$ is the entry-wise multiplications and the variables x_i and x_j are two random solutions. The parameter $\propto_0$ ranged from 0.01 up to 0.5 and the value 0.1 worked well for most of applications made by (Yang and Deb, 2011). Accordingly, when replacing the step length s calculated from Lévy Flights distribution, the step size is derived as below:

$$\text{step} = \propto_0 Levy(\beta)(x_i - x_j) \sim 0.1\ |s|^{-1-\beta}(x_i - x_j) \qquad \text{Equation 27}$$

The following approximation is used by (Xin-She Yang, SuashDeb, 2011):

$$\text{step} = \propto_0 Levy(\beta)(x_i - x_j) \sim 0.01 \frac{u}{|v|^{1/\beta}}(x_i - x_j) \qquad \text{Equation 28}$$

The steps of the CS are shown in pseudo code in Figure (24) and flow charted in Figure (25)are as follows(Yang and Deb, 2011):

1. Select values for CSA parameters which are the number of nests (n), the step size parameter (β), discovering probability (Pa), and maximum number of iterations for termination of the cycles.
2. Generate random initial population of n host nests (solutions) with objective function and d number of decision variables
3. Get a cuckoo randomly by means of Lévy flights
4. Evaluate its fitness so if the new fitness is better, it will replace the old one.
5. Abandon portion of worst nests based on a probability of (Pa) to decide whether a nest will keep its position
6. Rank solutions based on fitness and find the current best one.
7. Repeat the steps from 3-6 unless the stopping criteria satisfied.

Several comparisons made between the CS and other meta-heuristic algorithms such as GAs and PSO, and it is concluded that CS performs better than other algorithms. This is partly due to its fewer parameters needed to be adjusted (Yang and Deb, 2010).

Objective function $f(x)$, $x = (x_1, ..., x_d)^T$;
Initial a population of n host nests x_i $(i = 1, 2, ..., n)$;
while (t <MaxGeneration) or (stop criterion);
  Get a cuckoo (say i) randomly by Lévy flights;
  Evaluate its quality/fitness F_i;
  Choose a nest among n (say j) randomly;
  if $(F_i > F_j)$,
    Replace j by the new solution;
  end
  Abandon a fraction (p_a) of worse nests
    [and build new ones at new locations via Lévy flights];
  Keep the best solutions (or nests with quality solutions);
  Rank the solutions and find the current best;
end while
Postprocess results and visualisation;

Figure (24)

Single Objective Cuckoo Search (CS) Pseudo-code (Yang and Deb, 2010)

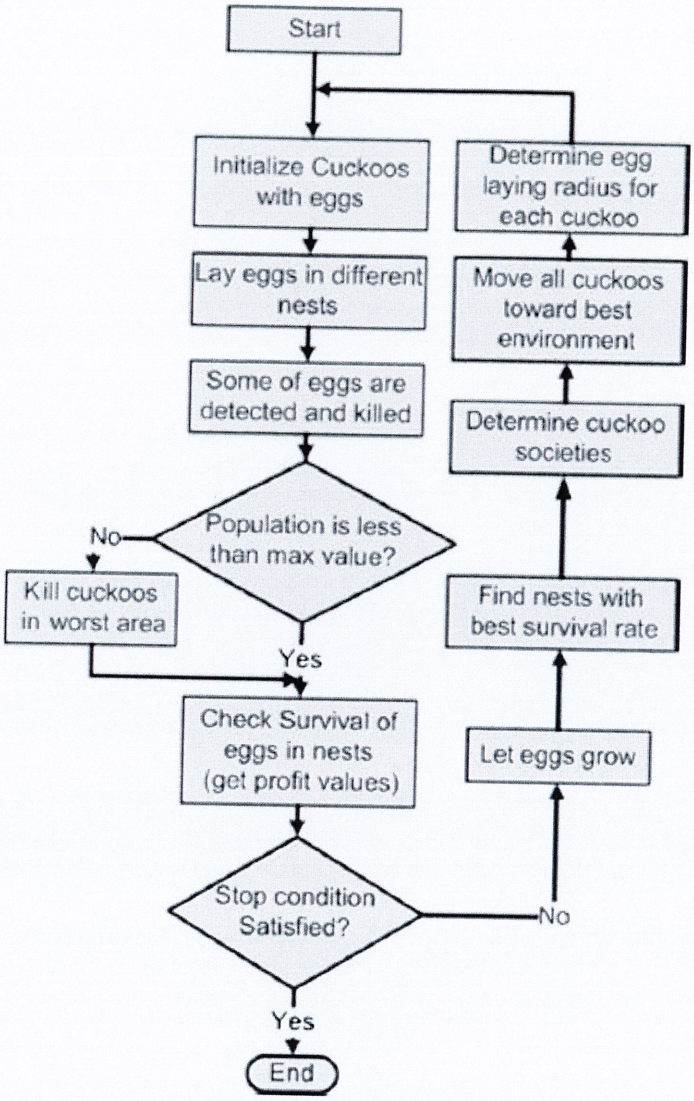


Figure (25)
Flow chart of CSA used by (Rajabioun, 2011)

Lévy Flights

The Lévy Flights are steps that transfer old solutions to new solutions based on random walk with step length drawn from Lévy Flight distribution. These steps are generated by Levy walk around old solutions which could be best solutions to create diversification and intensification. That is, this process intensifies potential solutions to solutions like best

solutions which may trap in a local optimum unless the substantial fraction of randomization diversifies it.

The Lévy distribution simple power law is calculated using step length s and step size parameter β as follows (Yang, 2010):

$$L(s) \sim u = |s|^{-1-\beta}. \qquad (0 < \beta \leq 2) \qquad \text{Equation29}$$

This relationship is basically formulated as below then derived into several special cases of step lengths as detailed by (Yang, 2010).

$$\mathrm{L}(\mathrm{s}, \gamma, \mu) = \begin{cases} \sqrt{\frac{\gamma}{2\pi}} exp\left[\frac{\gamma}{2(s-\mu)}\right] \frac{1}{(s-\mu)^{3/2}} & 0 < \mu < s < \infty \\ 0 & Otherwise \end{cases} \qquad \text{Equation 30}$$

Where π is called the location parameter, γ is the scaling parameter and the μ controls the scale of this distribution (Yang and Deb, 2011)

The step length s is calculated as follows:

$$s = \frac{u}{|v|^{1/\beta}} \qquad \text{Equation 31}$$

Where u and v are drawn from normal distributions. That is

$$u \sim N(0, \sigma_u^2), \; v \sim N(0, \sigma_v^2), \qquad \text{Equation 32}$$

With

$$\sigma_u = \left\{\frac{\Gamma(1+\beta)\sin(\pi\beta/2)}{\Gamma[(1+\beta)]\beta 2^{(\beta-1)/2}}\right\}, \sigma_v = 1 \qquad \text{Equation 33}$$

Here Γ is the standard Gamma function.

Various statistics are made to observe the behaviour of various animals and even human to develop the distribution of Lévy flights. This relationship is applied in meta-heuristics optimization and brought significant improvement in the solutions. To visualise the randomisation generated by the Lèvy Flight, (Yang, 2010) and (Yang and Deb, 2011) applied random step lengths to change the value of two solutions at once. This change is graphed in 2D sketch to illustrate the naturally inspired randomized steps. It is shown in 50 consecutive steps as in Figure (26), and 100 steps as shown in Figure (27). Also, the distribution of values obtained in 100 steps is shown in Figure (28) to illustrate the values of the step lengths.

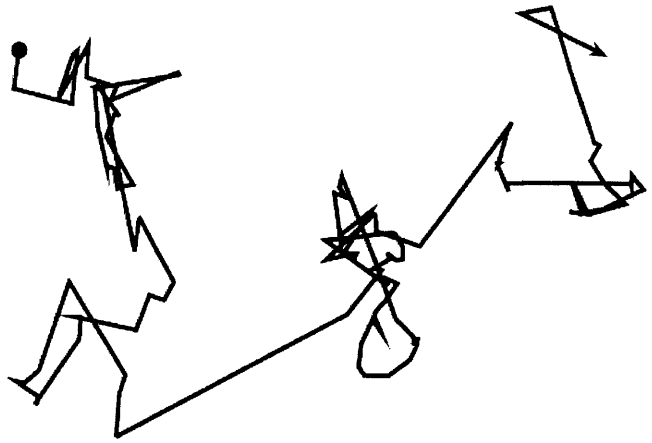


Figure (26)
2D Lévy flights in 50 steps (Yang, 2010)

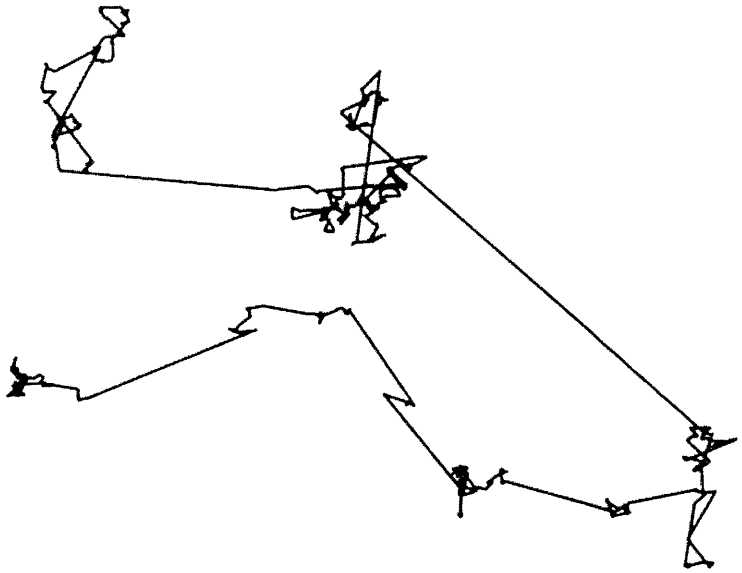


Figure (27)

2D Lévy flights in 100 steps (Yang and Deb, 2011)



Figure (28)

Distribution of 100 consecutive steps (Yang and Deb, 2011)

5.2.3 Multi-Objective Cuckoo Search Algorithm (MOCS)

The three guidelines used in CS are modified slightly to adapt with multi-objective optimization requirement and applied on the Pseudo-code shown in Figure (29) which is also converted into a flow chart in Figure (30). The rules used by (Yang and Deb, 2011) to write the algorithm when there is k number of objectives function:

1. A cuckoo bird lays k eggs- i.e. two eggs for two objectives each representing solutions for each objective - in random nests where k^{th} egg corresponds to the solution for k^{th} objective. This step aims to find new solution using a random walk or by Lévy flight in a way similar to crossover.
2. This rule is similar to that in single objective as the best quality nest(s) of better domination will remain in the next iterations to apply elitisms in a way that let the algorithm converge well.
3. The host bird will discover the eggs with probability of Pa with values range from 0 up to 1 and based on similarities/ differences of the eggs so when it discovers the cuckoo eggs it will abandon the nest and build a new one in a different location. For each nest, mutation is performed to discard the worst solutions and generate new solutions based on probability and the similarity of solutions to a solution. This step involves both of Lévy flights and differential of solutions to ensure the efficiency of the algorithm.

```
Initialize objective functions f1(x),...fk(x)          (x=x1,..xd)
Generate an initial population of n host nests xi and each with k eggs
 While(t<MaxGeneration) of (stop criterion)
        Get cuckoo (say i) randomly by lévy flights
        Evaluate and check if it is Pareto optimal
        Choose a nest among n (say j) randomly
        if new solutions of nest j dominate those of nest i,
              Replace nest i by the new solution set of nest j
        end
        Abandon a fraction (Pa) of worst nests
        Keep the best solutions (or nests with non-dominated sets)
              Sort and find the current Pareto optimal solutions
end
Post process results and visualization
```

Figure (29)
MOCS pseudo-code (Yang and Deb, 2011)

Step 1: initialize the MOCS factors which are step size parameter used to find lévy flight (β), discovering rate (Pa) used to eliminate nests, number of nests, and the termination criteria such as maximum number of iterations or tolerance if applicable. It is recommended by (Yang and Deb, 2011) to use a number of iterations ideally 5000 iterations $\beta = 1.5$ and Pa=0.5. Also he prefers using a fixed number of iterations than tolerance because it is easier in implementation and comparison of the Pareto front closeness in test problem. It is also important to initialize the number of nests (n) to be large to produce proper number of Pareto points and small to reduce computational time. Especially that program uses arrays to store the solutions in. In the paper of (Yang and Deb, 2011), they found that increasing the number of nests or running more time will

increase the Pareto points, but rerunning the program more times is more preferred due to the lower computation time.

Step 2: This step requires setting up a matrix of (n- rows) and (d-columns) which contains random solutions where d is the total number of decision variables and n is number of nests (number of solutions). After that, the objective functions are calculated for these solutions and the fitness is evaluated.

Step 3: In this step, a new solution is generated by means of random Lévy flights of cuckoo bird laying new eggs in the nests. A new solution is generated based on the old solution adding/ subtracting a random step size. That is, if we call the old solution as x^t and the new solutions $x^{(t+1)}$, then the new solution is measured by adding the step size of the cuckoo Lévy flight. This step is followed by applying simple bound to ensure that the solution does not go beyond the lower and upper bounds.

Step 4: Calculate the fitness or find the domination of the new solution so that if the new solutions are not worse in any objective function they will replace the old ones. This step serves to compare between the solutions to find whether they are Pareto using a tournament selection operator. Bearing in mind that solution vector u is considered dominating another vector $\boldsymbol{v}$ if and only if $u_i \leq v_i$ for $\forall\, i \in \{1, \ldots n\}$ and $\exists\, i \in \{1, \ldots n\} : u_i < v_i$;

Step 5: Abandon portion of worst nests based on a probability of (Pa). Then apply simple bound not to go beyond the lower and upper bounds.

Step 6: Evaluate the new solution, compare to current solution and keep the dominant solution by rules described in (Step 4). In addition, other Pareto optimums from previous iterations are considered as archived to be kept in mind.

Step 7: The previous steps are repeated until the maximum number of cycles is reached.

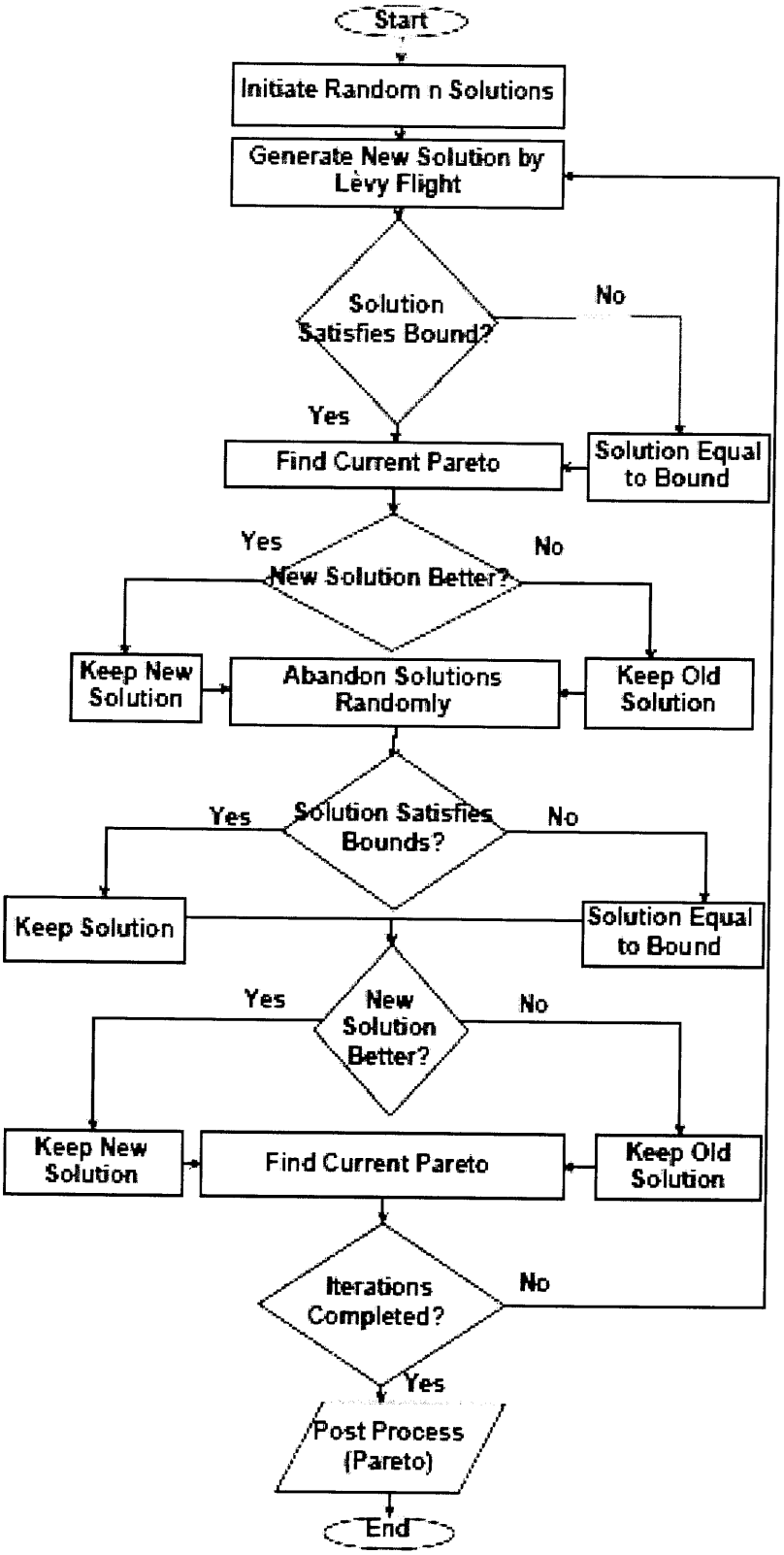


Figure (30)
MOCS Flow Chart

Illustrative Example

Problem Data:

Activity number	1	2	3
Material cost Rate	92	479	195

Quantity of material in each activity and unit

Quantities in activity	1	2	3
Unit1	1032	104	85
Unit2	1077	86	92

Productivity of available crew formations for each activity

	1	2	3	4
Activity1	89.77	71.81	53.86	
Activity2	5.73	6.88	8.03	
Activity3	9.9	8.49	7.07	5.66

Labor cost of available crew formations for each activity

	1	2	3	4
Activity1	3804	2853	1902	
Activity2	1875	2438	3000	
Activity3	3931	3238	2544	1850

Equipment cost of available crew formations for each activity

	1	2	3	4
Activity1	874	655	436	
Activity2	285	371	456	
Activity3	315	259	204	148

Decision Variable:

$xi \ where \quad i = 1,2,3 \ are \ crew \ formation \ number \ for \ activity \ 1,2,3$

$x4 \ interruption \ in \ activty \ 1 \ after \ unit \ 1, x5 \ \ interruption \ in \ activty \ 1$

$after \ unit \ 2, \ldots \ x9 \ interruption \ in \ activty \ 3 \ after \ unit \ 2$

Objective functions (equations 22 and 23)

Subjected to (shown in part 4.5)

Non-negativity:

$x(i) \geq 0 \quad i = 1,2,..,9$

The optimization is run with the following values:

	Nests n	Eggs k	Decision variables d	Iterations
Values	3	2	9	6

- Lower bounds = [1 1 1 0 0 0 0 0 0]
- Upper bounds =[3 3 4 10 10 10 10 10 10]

The run of MOCS results are as follows:

Step 1and 2: The initial nests with random values and fitness:

	x1	x2	x3	x4	x5	x6	x7	x8	x9	f1	f2
Nest1	2	2	2	4	10	9	5	7	4	56.8	584295.8
Nest2	3	2	3	10	2	3	6	8	9	69.8	585563.5
Nest3	2	2	1	9	2	3	7	8	1	60.3	601348.5

Step 3: Changing initial values using Lèvy Flights that randomly add or subtract values in decision variables as shown in nests 2 and 3:

New	x1	x2	x3	x4	x5	x6	x7	x8	x9	f1	F2
Nest1	2	2	2	4	10	10	9	10	4	56.8	584295.8
Nest2	3	2	4	10	2	3	7	8	10	73.0	579248.0
Nest3	2	2	1	10	2	6	10	10	1	61.3	604157.5

Step 4: Keeping current Pareto in initial nests by selecting solutions that are dominating more solutions by being not worse in any objective value:

	x1	x2	x3	x4	x5	x6	x7	x8	x9	f1	f2
1	2	2	2	4	10	9	5	7	4	56.8	584295.8

Updating nests with new solutions obtained in step 3 if new nests show at least one better objective value than that in the available nests:

	x1	x2	x3	x4	x5	x6	x7	x8	x9	f1	f2
Nest1	2	2	2	4	10	10	9	10	4	56.8	584295.8
Nest2	3	2	4	10	2	3	7	8	10	73.0	579248.0
Nest3	2	2	1	9	2	3	7	8	1	60.3	601348.5

Step 5: When random probability is higher than 50%, solutions are changed with random values such as in nest 1:

New	x1	x2	x3	x4	x5	x6	x7	x8	x9	f1	f2
Nest1	2	2	3	4	10	10	9	10	6	59.0	580187.3
Nest2	3	2	4	10	2	3	7	8	10	73.0	579248.0
Nest3	2	2	1	9	2	3	7	8	0	60.3	601348.5

Step 6: Updating nests with new solutions obtained in step 5 if new nests show at least one better objective value than that available in nests:

	x1	x2	x3	x4	x5	x6	x7	x8	x9	F1	f2
Nest1	2	2	3	4	10	10	9	10	6	59.0	580187.3
Nest2	3	2	4	10	2	3	7	8	10	73.0	579248.0
Nest3	2	2	1	9	2	3	7	8	0	60.3	601348.5

Finding Pareto solutions in nests obtained in step 6 and previous Pareto - in step 4- by selecting solutions that dominate more of the other solutions:

	x1	x2	x3	x4	x5	x6	x7	x8	x9	f1	f2
1	2	2	2	4	10	9	5	7	4	56.8	584295.8
2	2	2	3	4	10	10	9	10	6	59.0	580187.3
3	3	2	4	10	2	3	7	8	10	73.0	579248.0

Step 7: These obtained Pareto solutions are plotted after six iterations which is done when each of the three nests is updated twice.

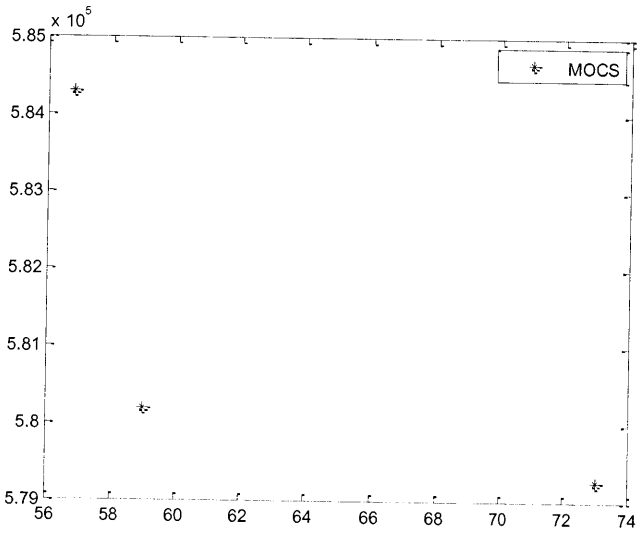


Figure (31)
Pareto Front of the illustrating example

Efficiency of MOCS

There are several studies and applications where the efficiency of CSA is demonstrated. That's to say; it is proved that the CS satisfies the requirements of global convergence unlike the convergence of PSO to local optimum in multimodal optimization which is controlled by the discovery probability. That is for discovery rate of Pa = 0.25, the convergence to local is made by 1/4 of the search time while the rest of time is for the global search that allows to explore well on the global scale. Another lead of the CS is the use of Lévy Flights which have infinite mean and variance, which let the algorithm explore the search space more efficiently than standard Gaussian process (Yang and Deb, 2013).

The advantages of this algorithm are that it depends on few numbers of parameters which are the population size n, and the parameter Pa. Besides, the CS algorithm depends on three components: selection of the best solution as equivalent to elitism, exploitation by local random walk, and exploration by randomization via Lévy flights globally. However, it is important to bear in mind when the step of Lévy flights is larger it could be more efficient unless if there is possibility to move too far. The simulations made by (Gandomi et. al, 2011) found that CS is "more generic and robust for many optimization problems, compared with other meta-heuristic algorithms" due to its convergence rate which is insensitive to the parameter Pa. In other words, this insensitivity means there is no need to adjust the parameters (n and Pa) for each problem in consideration. Moreover, the CS algorithm is used in solving engineering problems especially after its proof of solving benchmark problems. (Gandomi et. al, 2011) There are important benchmark test problems that

are used to evaluate whether the algorithm can find the Pareto front despite of the challenges in the test problem. Also, analysts are concerned with the accuracy and speed of the algorithm which is important to be reported as the error per number of iterations.

Schaffer's Min–Min (SCH)

The decision variable x is bounded between [-1000, 1000] in this problem. This test function is special for its convex Pareto front with the following two objective functions:

$$f_{1,}(x) = x^2$$

$$f_{2,}(x) = (x-2)^2$$

The resulted Pareto-front is diverse and converging to the correct curve when it was solved by the MOCS. The result obtained by (Yang and Deb, 2011) is shown in Figure (32) which is same as the one tested to approve validity in the study as shown in Figure (33). The MOCS that optimised the functions is shown in Appendix (1)

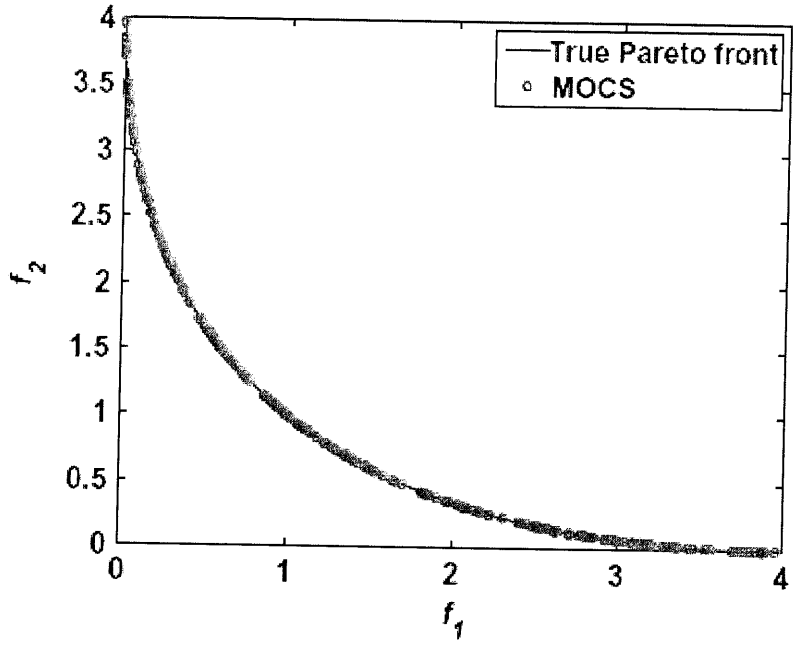


Figure (32)

Pareto fronts of test functions SCH (Yang and Deb, 2011)

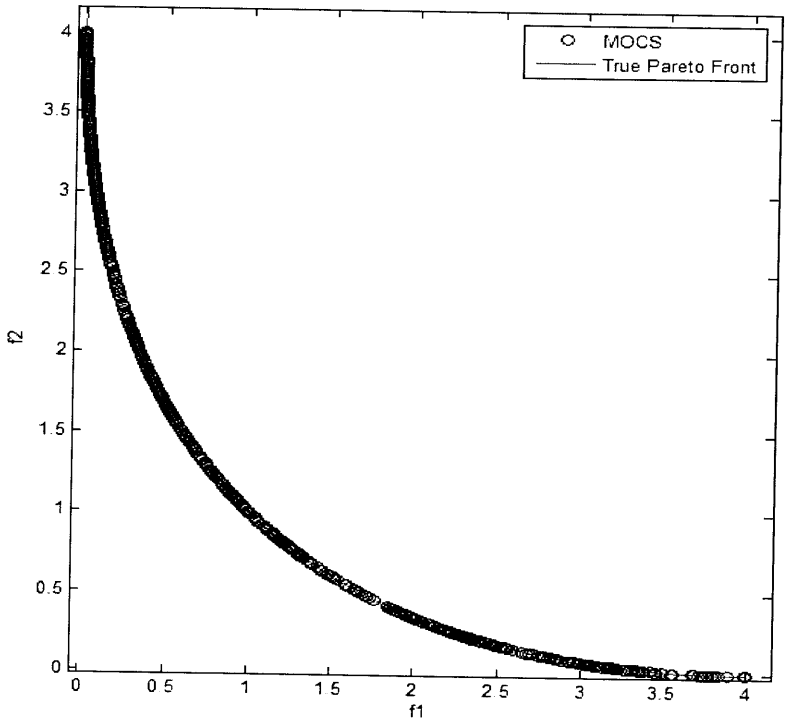


Figure (33)

Obtained Pareto fronts of test functions SCH in this study

Test problem ZDT1

This function is challenging because it contains numerous variables ($d = 30$) that results with a convex Pareto front of the following two objectives:

$$f_{1,}(x) = x_{1,}$$

$$f_2(x) = g(1 - \sqrt{f_{1,}/g})$$

$$g = 1 + \frac{9\sum_{i=2}^{d} x_i}{d-1}, x_i \in [0,1], i = 1, \cdots, 30,$$

The Pareto optimality is reached when $g = 1$ so that the true Pareto front is calculated as$f_2 = (1 - \sqrt{f_{1,}})$. The Pareto-front obtained diverse and converging to the correct curve when it was solved by the MOCS. The result obtained by (Yang and Deb, 2011) is shown in Figure (34) which is same as the one tested to approve validity in the study as shown in Figure (35). The MOCS that optimised the functions is shown in Appendix (1)

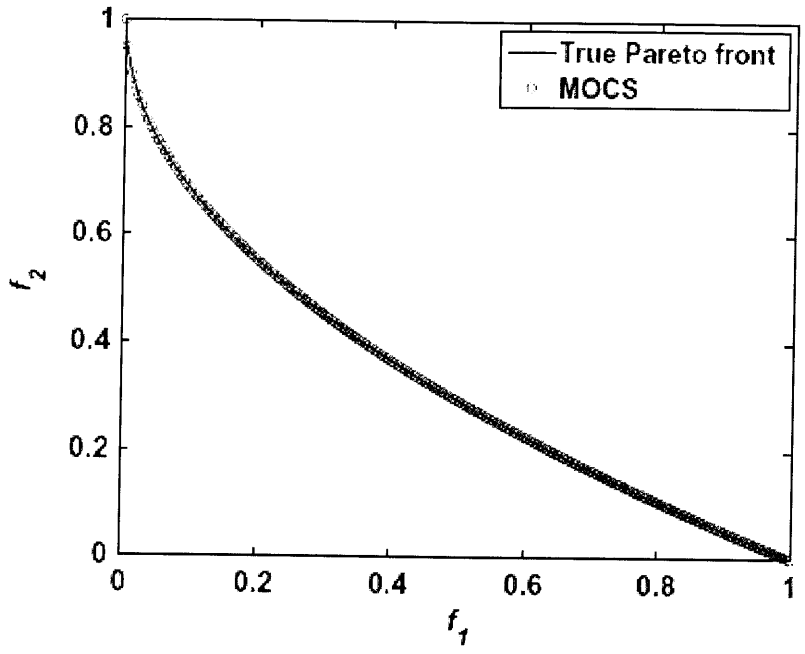


Figure (34)
Pareto fronts of test functions ZDT1 (Yang and Deb, 2011)

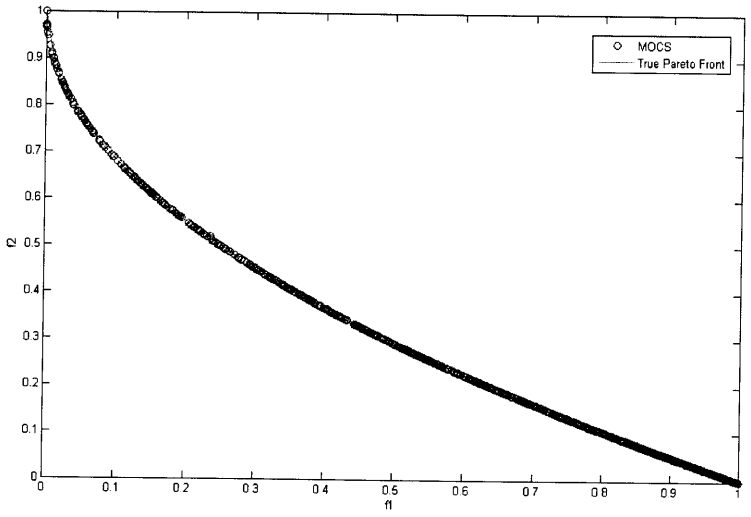


Figure (35)
Obtained Pareto fronts of test functions ZDT1 in this study

Test problem ZDT2 function with a non-convex front (Yang and Deb, 2011)

$$f_{1,}(x) = x_{1,}$$

$$f_2(x) = g(1 - \frac{f_{1,}}{g})^2,$$

$$g = 1 + \frac{9\sum_{i=2}^{d} x_i}{d-1}, x_i \in [0,1], i = 1, \cdots, 30,$$

Where $d = 30$ the number of variables and the Pareto optimality is is reached when $g = 1$ so that the true Pareto front is calculated as $f_2 = (1 - f_{1,})^2$. The Pareto-front obtained diverse and converging to the correct curve when it was solved by the MOCS. The result obtained by (Yang and Deb, 2011) is shown in Figure (36) which is same as the one tested to approve validity in the study as shown in Figure (37). The MOCS that optimised the functions is shown in Appendix (1)

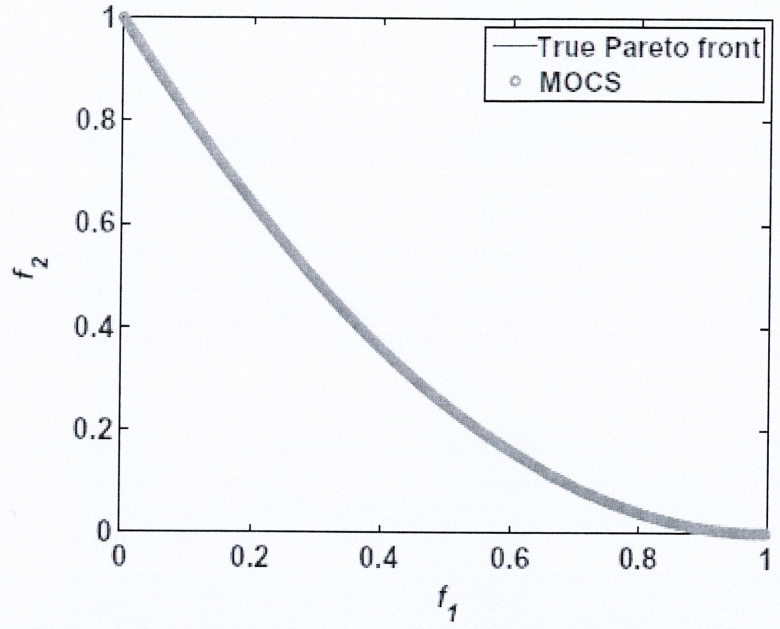


Figure (36)

Pareto fronts of test functions ZDT2 (Yang and Deb, 2011)

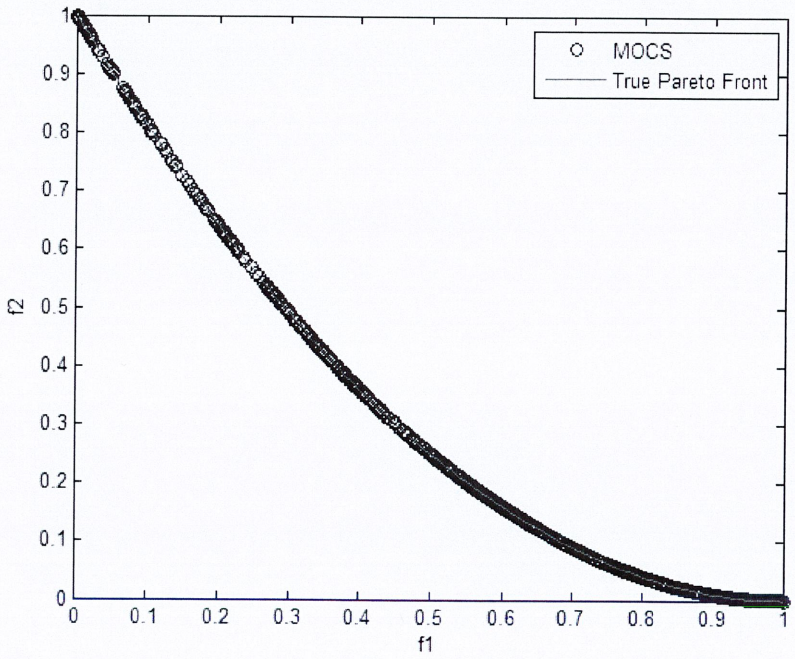


Figure (37)

Obtained Pareto fronts of test functions ZDT2 in this study

Test problem ZDT3 function with Pareto of noncontiguous convex parts (Yang and Deb, 2011)

$$f_{1,}(x) = x_{1,}$$

$$f_2(x) = g\left[1 - \sqrt{\frac{f_1}{g}} - \frac{f_1}{g}\sin(10\pi f_1)\right]$$

$$g = 1 + \frac{9\sum_{i=2}^{d} x_i}{d-1}, x_i \in [0,1], i = 1, \cdots, 30,$$

Where $d = 30$ is the number of dimensions (variables). The Pareto optimality is reached when $g = 1$ so that the true Pareto front is calculated as $f_2 = \left[1 - \sqrt{f_1} - f_1 \sin(10\pi f_1)\right]$. The Pareto-front shape as obtained in Figure (38) discontinuity in the Pareto- front due to the sine function. However, obtained diverse and converging to the correct curve when it was solved by the MOCS. The MOCS that optimised the functions is shown in Appendix (1)

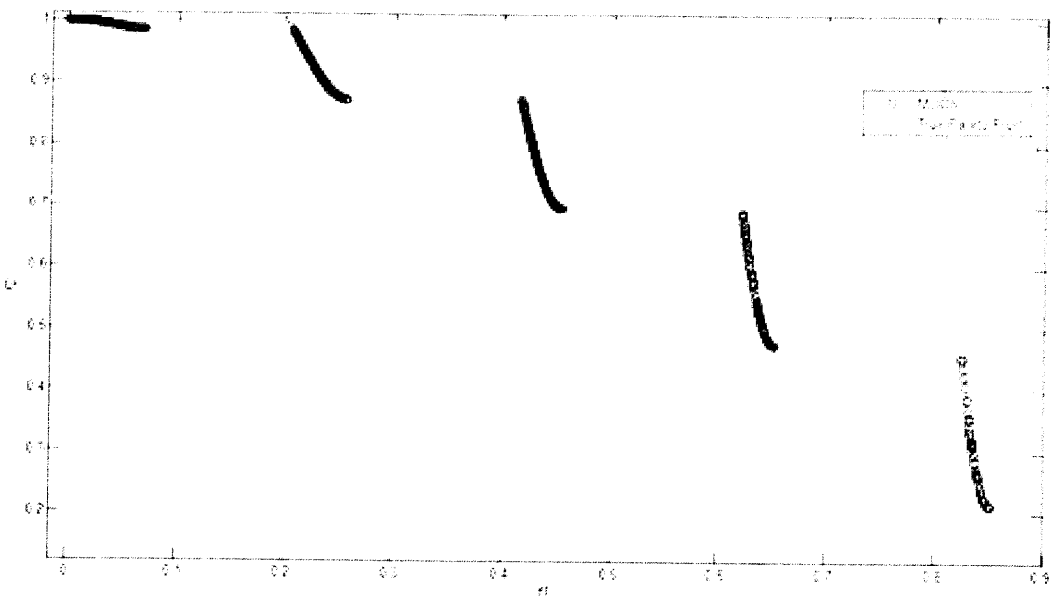


Figure (38)

Obtained Pareto fronts of test functions ZDT3

Chapter 6

Experimentations and Results

6.1 Computational Applications:

- Case Study 1
- Case Study 2

Chapter 6

Experimentations and Results

6.1 Computational Applications

The problems to be solved in this chapter are repetitive in nature which makes each activity repeated throughout all units or segments. The purpose of optimisation is to find best crew formation for each activity and best interruption days to trade-off time and cost for the project. The MOCS is used to optimise a repetitive construction projects to introduce this method for decision makers. The MAT-LAB codes used in optimization is shown in Appendix (2)

6.1.1 Time-Cost Trade-off Case studies

In this part, two case studies are studied. The first one is three span bridge construction problems taken from (Hyari et. al, 2009). The problem is for a three spans concrete bridge that contains four repeating parts. The project includes five main activities in its construction which are excavation, foundation, columns, beams and slabs. The second case study is selected from one of the construction projects executed in Bahrain. This project is treated slightly differently as the search for the data came from several contactors. Those contractors are at the same level of competence in the eyes of ministry of housing who could select any one of them. However, the criterion that the Ministry uses in the selection of contractors is only based on the cost per square meters.

There are two approaches which are used by the ministry of housing and other ministries to make a decision in their repetitive projects. The first approach is to search for the best crew formations amongst all the available data assuming that the contractor has alternative resources. The second approach is to search for the best contractor among contractors that trade-off the time and cost.

The time-cost model calculates the total duration of the project and the direct cost when assigning a set of decision variables. The duration is calculated based on equality and inequality constraints to satisfy logical relationships. Therefore, when the MOCS is used, it obtains Pareto optimal solutions in a Pareto optimum curve, as a result decision makers then can choose one solution based on higher level of information.

Efficiency of Time-Cost Module

The efficiency of time-cost module is affected by the accuracy of the data given by contractors which could be enhanced when the contractor are obliged to give correct data for the sake of winning the contract. The data contained the productivities in cubic meters per day per activity with the corresponding labour cost and equipment cost. Since there is several ranges of productivities, only two speeds of productions are requested from each contractor: which are normal speeds and fast speed. The contractors do not usually give the method statements as it is one of their secrets and could be one of their competencies in the market, so they barely give the data in numbers as shown in Appendix (3). However, if the ministry adapt time and cost trade-off, the contractors possibly pay more efforts to submit realistic data. For the reason that the ministry assigns the contractor with crew formations that trade-off time and cost.

Therefore, when the contractor exaggerates the prices of the crew formations or misestimates the productivities, they will not win the contract. In the other hand, when a contractor reduce the prices or overestimate their productivities, they may fail in meeting time and cost during the project execution. That is why, when the data are requested by ministry top engineers, the contractors becomes more cooperative in giving realistic data. In addition, only the commonly contacted contractors with the ministry of housing are requested to ensure their data are commonly dealt with in the ministry projects. One of the recommendations is that it is better for the ministry to collect such data from time to time to enhance their hiring of contractors. It worth pointing out that the project in Bahrain is done in the actual life, so this research is made to find a better solution considering the actual solution made by the ministry of housing. However, this research considers only a sample of important activities and not inclusive of the whole project. This makes the term "project" given only for the activities in considerations and not all activities considered in the contract.

The model used in the case studies is written in MAT-LAB in a way that makes it easy to adjust the parameters, the number of activities and the number of repetitive units as shown in in Appendix (4) and Appendix (5). So that, when modelling case study 2, the only changes taken into consideration are the parameters used. Otherwise, in the formulation used in case study 1 assigns the method to the activists by means of factors in a tedious way as illustrated in case study 1.

Case study 1: Three Span Bridge

This case study is considered in order to compare the results obtained by MOCS with the results of (Hyari et. al, 2009). This project is three-span concrete bridge consists of four repetitive sections, each containing five construction activities: excavation, foundations, columns, beams and slabs. The scheduling constraint assumed is the precedence relationships with no lag time. That is, the successive activities start immediately after the finishing of the predecessor activity unless interruptions occur. One of the very important parameter in the time-cost model is the quantities of work for the five activities through all the four repetitive units. The second parameter includes the available crew formation options or crew formation available in terms of productivity rates and daily direct costs. This parameter is the clue from which crew options are differentiated in time and cost. That is, when the project is changed, only these parameters should be changed to let the model work for other quantities and crew formations.

The formulation is based on the fact that there are specific number of activities and number of repetitive units. As shown in Appendix (4) and Appendix (5), the productivities and costs are assigned in matrices that consider the column numbers and row numbers as chosen by the decision variables. For example, the quantities Q are organized as in Table (1) with 5 rows and four columns so that when the decision variable is dealing with unit 1 activity 1 the quantity used is Q (1, 1) from the matrix. Furthermore, Table (2) shows productivity of activities in rows and the crew formations or crew formations in columns so that when the decision variable selects first crew formation for excavation the productivity is P (1, 1). That is, when calculating the duration of the first unit of the first

activity using method1, Q (1, 1) is divided by P (1, 1). This method avoids lengthy modelling which could result in miscalculations that are not easy to follow. Besides the fact that it makes it more flexible to change the number of activities, number of units, and number of crew formations only by changing the matrices rows and columns. In the other hand, when writing the formulation without matrices, there should be some factors that are considered to make the decision variables keep only the productivities of the selected method and delete unselected methods. In this section the formulation is written in a tedious way for illustration as it is possible to write it in this way due to the small number of units which is only four. In the next case study, however, the number of units is 156 which make it very lengthy and impractical to write it with factors.

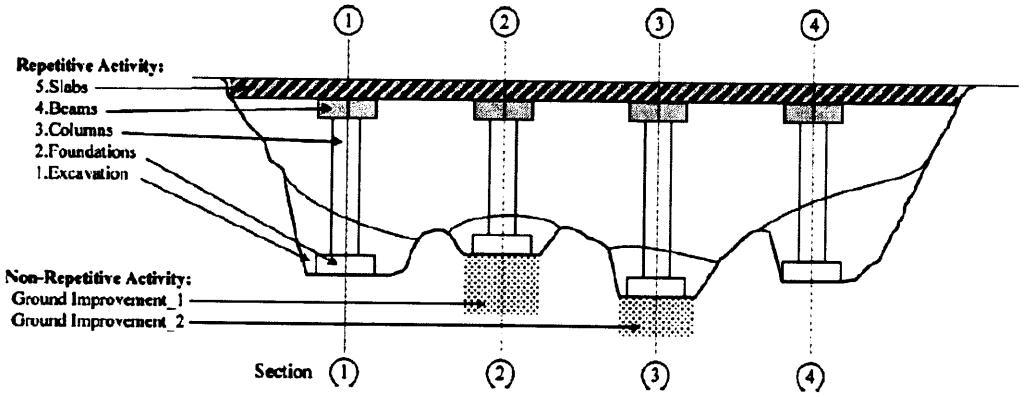


Figure (39)
The bridge case study (Liu and Wang , (2007)

Higher level of information

Higher level of information is those additional data used to support decision makers' selection of one final answer. This part is considered after obtaining the Pareto front as per the information used by (Hyari et al, 2009). This includes the priority of cost objective function, the indirect

costs of the daily construction work, liquidated damages (LDC.), and the completion incentives. They are related to the project duration D obtained from the Pareto front.

There are four rates taken from (Hyari et al, 2009) included in four calculations as following:

1. ICR= $2500 which is the indirect cost rate per day of the project.
2. DCI=$5000 which is the daily incentives for early completion paid to the contractor
3. DLD= $8000 which is the daily liquidated damages paid by the contractor per day later than SD days.
4. LDRC = $1000 which is the lane daily rental cost paid by contractor per every day in the project.

Where D is the scheduled project duration in days and SD is the duration of 115 as specified by the contract. The following calculations are made to find other costs:

Indirect costs:

$$IC = ICR \cdot D$$

$$IC = 2500 \cdot D$$

Early Completion Incentives

$$ECI = DCI \cdot (SD - D)$$

$$ECI = 5000 \cdot (SD - D)$$

Liquidated Damages

$$LDC = DLD \cdot (D - SD)$$

$$LDC = 8000 \cdot (D - SD)$$

Lane Rental Cost

$$LRC = LDRC.\ D$$

$$LRC = 1000.\ D$$

Results

The results obtained using MOCS are better results in terms of diversity and number of solutions. That is, the number of different solutions obtained in case study 1 by (Hyari et. al, 2009) using GA was 24 points as shown in Table (5). In the MOCS, however, the number of Pareto points is 42 points of different solutions as shown in Figure (40), and Table (6). Nevertheless, these 42 solutions consider the decimal numbers in days. If the solutions considered as integers with no decimal points then there are 25 different solutions obtained by MOCS as shown in Figure (41).

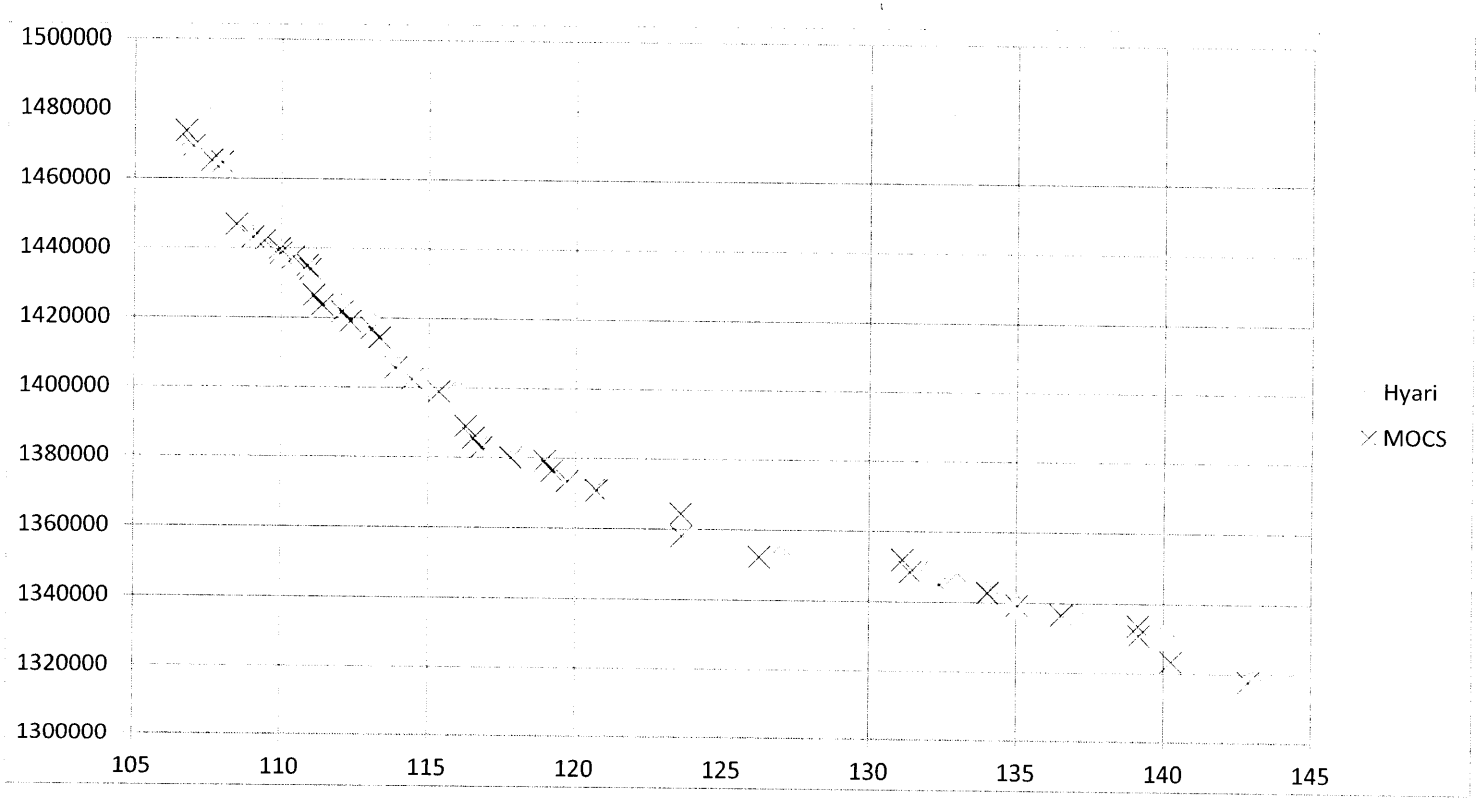


Figure (40)

Pareto front of MOCS and the one obtained by (Hyari et. al, 2009)

Table (5)
Solutions obtained for three span concrete bridge by (Hyari et. al, 2009)

Solution (s)	Project performance		Selected crew formation (n_i*)					Total interruption days
	Project duration in days (D)	Project direct cost ($)	Excavation (i = 1)	Foundation (i = 2)	Columns (i = 3)	Beams (i = 4)	Slabs (i = 5)	
1	107	1 469 361	1	1	3	1	1	14
2	108	1 465 115	1	1	3	1	1	13
3	109	1 443 297	1	1	3	2	1	10
4	110	1 438 619	1	1	3	2	1	9
5	111	1 433 941	1	1	3	2	1	8
6	112	1 423 748	1	1	3	3	1	7
7	113	1 419 070	1	1	3	3	1	6
8	114	1 405 850	1	2	3	2	1	5
9	115	1 402 353	1	2	3	2	1	4
10	116	1 398 856	1	2	3	2	1	3
11	117	1 382 803	1	2	3	3	1	1
12	118	1 380 055	1	2	3	3	1	0
13	120	1 373 543	1	2	2	3	1	1
14	121	1 370 795	1	2	2	3	1	0
15	124	1 358 021	1	2	2	4	1	0
16	127	1 352 263	1	2	2	4	2	0
17	132	1 348 405	1	2	1	4	1	1
18	133	1 344 897	1	2	1	4	1	0
19	134	1 342 647	1	2	1	4	2	1
20	135	1 339 139	1	2	1	4	2	0
21	137	1 336 524	1	3	2	4	1	0
22	140	1 330 766	1	3	2	4	2	0
23	141	1 323 400	1	3	1	4	1	0
24	143	1 317 642	1	3	1	4	2	0

Table (6)
Solutions obtained by the MOCS

Solutions	Project Performance		Selected Crew formation (n_i^*)					Total Interruptions
	Project Duration	Project Direct Cost ($)	Excavation	Foundation	Columns	Beams	Slabs	
1	106.8	1473606.7	1	1	3	1	1	15
2	107.0	1469360.7	1	1	3	1	1	14
3	107.6	1465114.7	1	1	3	1	1	13
4	108.0	1464682.7	1	1	3	1	1	13
5	108.5	1446794.4	1	1	3	2	1	11
6	109.0	1443297.4	1	1	3	2	1	10
7	109.4	1442116.4	1	1	3	2	1	10
8	109.9	1439800.4	1	1	3	2	1	9
9	110.0	1438619.4	1	1	3	2	1	9
10	110.4	1437438.4	1	1	3	2	1	9
11	110.9	1435122.4	1	1	3	2	1	8
12	111.0	1433941.4	1	1	3	2	1	8
13	111.1	1426495.6	1	1	3	3	1	8
14	111.4	1423747.6	1	1	3	3	1	7

15	112.1	1421817.6	1	1	3	3	1	7
16	112.3	1419069.6	1	1	3	3	1	6
17	113.1	1417139.6	1	1	3	3	1	6
18	113.3	1414391.6	1	1	3	3	1	5
19	113.9	1405850.1	1	2	3	2	1	5
20	114.4	1402353.1	1	2	3	2	1	4
21	115.4	1398856.1	1	2	3	2	1	3
22	116.3	1389007.3	1	2	3	3	1	3
23	116.5	1385551.3	1	2	3	3	1	2
24	116.8	1382803.3	1	2	3	3	1	1
25	117.8	1380055.3	1	2	3	3	1	0
26	119.0	1379038.9	1	2	2	3	1	3
27	119.2	1376290.9	1	2	2	3	1	2
28	119.7	1373542.9	1	2	2	3	1	1
29	120.7	1370794.9	1	2	2	3	1	0
30	123.6	1364338.1	1	2	2	4	1	
31	123.6	1358021.1	1	2	2	4	1	0
32	126.3	1352262.8	1	2	2	4	2	0

33	131.1	1351913.4	1	2	1	4	1	2
34	131.4	1348405.4	1	2	1	4	1	1
35	132.4	1344897.4	1	2	1	4	1	0
36	134.0	1342647.1	1	2	1	4	2	1
37	135.0	1339139.1	1	2	1	4	2	0
38	136.5	1336524.0	1	3	2	4	1	0
39	139.1	1333574.7	1	3	2	4	2	1
40	139.2	1330765.7	1	3	2	4	2	0
41	140.3	1323400.3	1	3	1	4	1	0
42	142.9	1317642.0	1	3	1	4	2	0

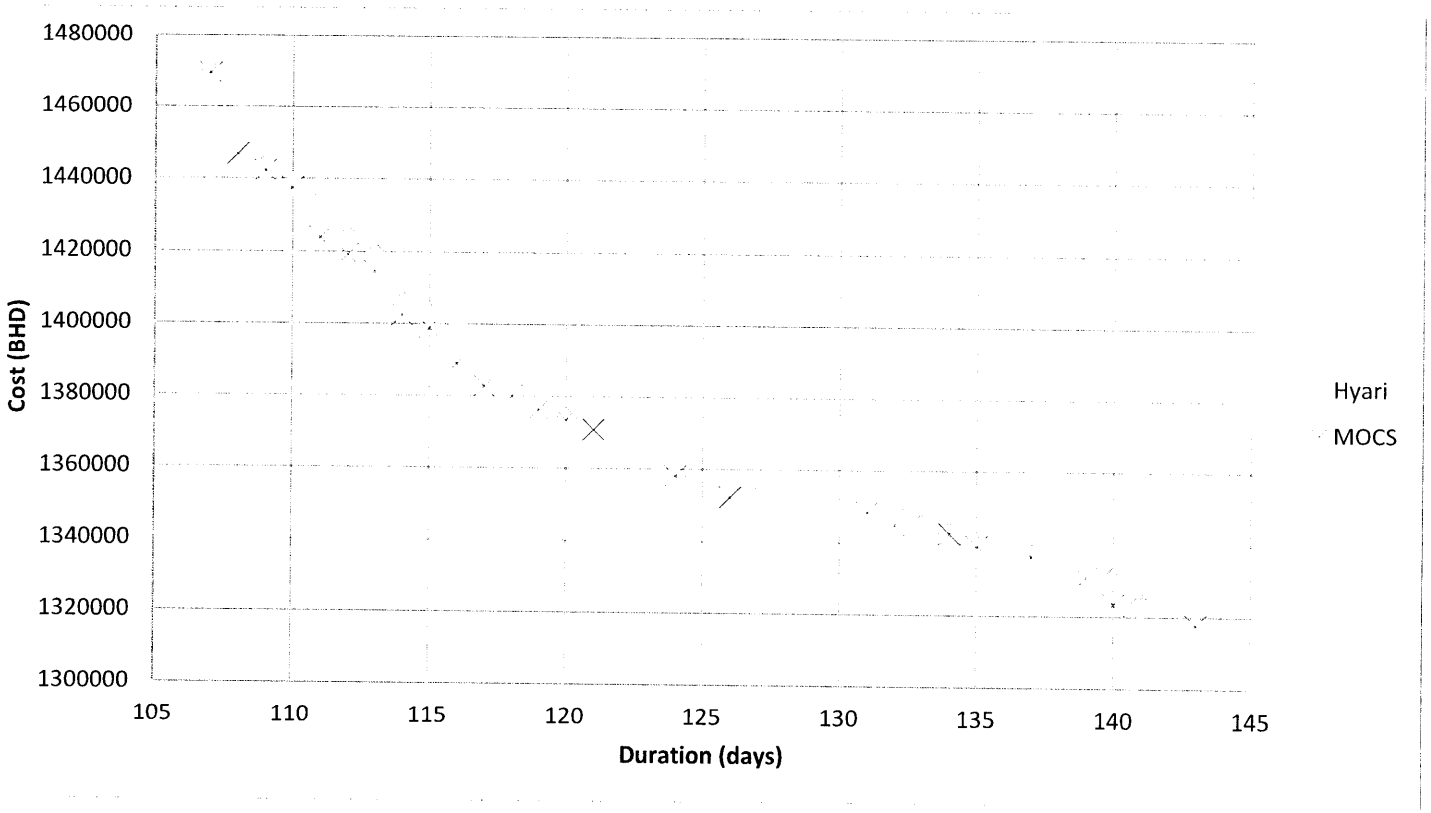


Figure (41)
Comparison between rounded results obtained by MOCS and by (Hyari et. al, 2009)

The results obtained by (Hyari et. al, 2009) are extended to include the indirect costs beside the Pareto solutions which eventually ended up with finding solely one optimum solution. These solutions are calculated in terms of the indirect costs to find the minimum total cost as shown in Table (7). From Table (8), it is clear that the results of MOCS are better than that of (Hyari et. al, 2009) in the first two scenarios. That is, in scenario 1 when subtracting the completion incentives for early completion of project the cost will reduce. The MOCS found that $1660794 is the minimum while (Hyari et. al, 2009) found $1667797 as the minimum value. In the second scenario, when adding liquidated damages for late completion of project the cost will increase. The MOCS found less cost of $1660794 to be the minimum while in (Hyari et. al, 2009) found $1,685,303 to be the minimum. The final scenario is considering the lane rental costs added to increase the cost. The total cost in the last column showed the minimum cost is to be $1792021 which is the same in the MOCS and the study made by (Hyari et. al, 2009).

Table (7)

Total costs of solutions obtained by (Hyari et. al, 2009).

	Project performance				Scenario 1		Scenario 2		Scenario 3	
Solution (s)	Project duration in days (D)	Project direct cost ($)	Project indirect cost ($)	Project total cost ($)	Completion incentives ($)	Project total cost ($)	Liquidated damages ($)	Project total cost ($)	Lane rental fee ($)	Project total cost ($)
1	107	1 469 361	267 500	1 736 861	64 000	1 672 861		1 736 861	107 000	1 843 861
2	108	1 465 115	270 000	1 735 115	56 000	1 679 115		1 735 115	108 000	1 843 115
3	109	1 443 297	272 500	1 715 797	48 000	1 667 797		1 715 797	109 000	1 824 797
4	110	1 438 619	275 000	1 713 619	40 000	1 673 619		1 713 619	110 000	1 823 619
5	111	1 433 941	277 500	1 711 441	32 000	1 679 441		1 711 441	111 000	1 822 441
6	112	1 423 748	280 000	1 703 748	24 000	1 679 748		1 703 748	112 000	1 815 748
7	113	1 419 070	282 500	1 701 570	16 000	1 685 570		1 701 570	113 000	1 814 570
8	114	1 405 850	285 000	1 690 850	8000	1 682 850		1 690 850	114 000	1 804 850
9	115	1 402 353	287 500	1 689 853		1 689 853		1 689 853	115 000	1 804 853
10	116	1 398 856	290 000	1 688 856		1 688 856	5000	1 693 856	116 000	1 804 856
11	117	1 382 803	292 500	1 675 303		1 675 303	10 000	1 685 303	117 000	1 792 303
12	118	1 380 055	295 000	1 675 055		1 675 055	15 000	1 690 055	118 000	1 793 055
13	120	1 373 543	300 000	1 673 543		1 673 543	25 000	1 698 543	120 000	1 793 543
14	121	1 370 795	302 500	1 673 295		1 673 295	30 000	1 703 295	121 000	1 794 295
15	124	1 358 021	310 000	1 668 021		1 668 021	45 000	1 713 021	124 000	1 792 021
16	127	1 352 263	317 500	1 669 763		1 669 763	60 000	1 729 763	127 000	1 796 763
17	132	1 348 405	330 000	1 678 405		1 678 405	85 000	1 763 405	132 000	1 810 405
18	133	1 344 897	332 500	1 677 397		1 677 397	90 000	1 767 397	133 000	1 810 897
19	134	1 342 647	335 000	1 677 647		1 677 647	95 000	1 772 647	134 000	1 811 647
20	135	1 339 139	337 500	1 676 639		1 676 639	100 000	1 776 639	135 000	1 811 639
21	137	1 336 524	342 500	1 679 024		1 679 024	110 000	1 789 024	137 000	1 816 024
22	140	1 330 766	350 000	1 680 766		1 680 766	125 000	1 805 766	140 000	1 820 766
23	141	1 323 400	352 500	1 675 900		1 675 900	130 000	1 805 900	141 000	1 816 900
24	143	1 317 642	357 500	1 675 142		1 675 142	140 000	1 815 142	143 000	1 818 142

Table (8)

Total costs of solutions obtained by MOCS

	Performance		Scenario1		Scenario2		Scenario3	
solution	Project Indirect Cost ($)	Project Total Cost ($)	Completion Incentives	Total Cost ($)	Liquidated Damages($)	Total Cost ($)	Lane Rent fees ($)	Project Total Cost ($)
1	267500	1741107	64000	1677107		1741107	107000	1848107
2	267500	1736861	64000	1672861		1736861	107000	1843861
3	270000	1735115	56000	1679115		1735115	108000	1843115
4	270000	1734683	56000	1678683		1734683	108000	1842683
5	270000	1716794	56000	**1660794**		1716794	108000	1824794
6	272500	1715797	48000	1667797*		1715797	109000	1824797
7	272500	1714616	48000	1666616		1714616	109000	1823616
8	275000	1714800	40000	1674800		1714800	110000	1824800
9	275000	1713619	40000	1673619		1713619	110000	1823619
10	275000	1712438	40000	1672438		1712438	110000	1822438

11	277500	1712622	32000	1680622		1712622	111000	1823622
12	277500	1711441	32000	1679441		1711441	111000	1822441
13	277500	1703996	32000	1671996		1703996	111000	1814996
14	277500	1701248	32000	1669248		1701248	111000	1812248
15	280000	1701818	24000	1677818		1701818	112000	1813818
16	280000	1699070	24000	1675070		1699070	112000	1811070
17	282500	1699640	16000	1683640		1699640	113000	1812640
18	282500	1696892	16000	1680892		1696892	113000	1809892
19	285000	1690850	8000	1682850		1690850	114000	1804850
20	285000	1687353	8000	1679353		1687353	114000	1801353
21	287500	1686356	0	1686356	0	1686356	115000	1801356
22	290000	1679007		1679007	5000	**1684007**	116000	1795007
23	292500	1678051		1678051	10000	1688051	117000	1795051
24	292500	1675303		1675303	10000	1685303*	117000	1792303
25	295000	1675055		1675055	15000	1690055	118000	1793055
26	297500	1676539		1676539	20000	1696539	119000	1795539
27	297500	1673791		1673791	20000	1693791	119000	1792791
28	300000	1673543		1673543	25000	1698543	120000	1793543

29	302500	1673295		1673295	30000	1703295	121000	1794295
30	310000	1674338		1674338	45000	1719338	124000	1798338
31*	310000	1668021*		1668021	45000	1713021	124000	**1792021***
32	315000	**1667263**		1667263	55000	1722263	126000	1793263
33	327500	1679413		1679413	80000	1759413	131000	1810413
34	327500	1675905		1675905	80000	1755905	131000	1806905
35	330000	1674897		1674897	85000	1759897	132000	1806897
36	335000	1677647		1677647	95000	1772647	134000	1811647
37	337500	1676639		1676639	100000	1776639	135000	1811639
38	342500	1679024		1679024	110000	1789024	137000	1816024
39	347500	1681075		1681075	120000	1801075	139000	1820075
40	347500	1678266		1678266	120000	1798266	139000	1817266
41	350000	1673400		1673400	125000	1798400	140000	1813400
42	357500	1675142		1675142	140000	1815142	143000	1818142

Bolt font for project minimum total cost and * for the project minimum total cost by (Hyari et. al, 2009)

Case study 2: Housing Project

In this case study, a housing project made by the ministry of housing is considered for planning in terms of direct cost and schedule. The housing project is in Darkulaib and consists of construction of 156 villas, 93 of them are of type "T3M" and 63 of type "D5" where T3M and D5 are the code names according to the Ministries' standards. Type T3M - Double Story villa -with Area of 208.96m^2- contains 3 Bedrooms, 2 Halls, 1 Family Hall, 1 Majlis, 1 Kitchen, 1 Toilet, and 2 Bathrooms. The second type is Type D5 which is also Double Story, but with slight difference in design with Area of 197.76m^2, it contains, 3 Bedrooms, 1 Family Hall, 1 Majlis, 1 Kitchen, 1Toilet, 2 Bathrooms, and 1 Store. The two types of villas are with staircase room, covered Garage, and boundary walls. The city located in Darkulaib where preparation and earthworks took place. More details about the contract are found in Appendix (6). Within the project only five activities are considered which are excavation, foundation, columns, beams, and slabs. The units of T3M are required to be completed to start units D5. The data about these activates is placed in Appendix (7) and the data given by project contractor in Appendix (3).

Case Study 2A: Considering One Contractor

In this case the three contractors from whom the data are given in Appendix (3) are considered as one contractor who can have different degrees in productivities. These contractors are all class 'A' contractors and are considered as good in management and innovation in the point view of the ministry. In other words, this is to assume that the contractor can hire any crew formations from the ones in the table by means of

subcontracting for some activities. This yields three matrices like that in case study 1 bearing in mind that they should be ranked according to productivities and costs so that they can be meaningful in trade-off. In other words, the crew formations with higher productivity and higher cost come later in the matrix and those with less productivity and higher cost are eliminated.

Parameters

The only difference in the formulation of this case study from case study 1 is the use of different parameters which are the quantities, activates crew formations, direct costs of labour and equipment. These parameters are arranged in Table (9) to Table (12). The arrangement considers ranking of the productivities so that the higher the productivities the more costly it is in either labour or equipment.

Table (9)
Quantity of materials in cubic meters per each unit

Activities/Units	Unit1	Unit2	Unit3	Unit4	Unit5
Excavation	5	5	5	6	6
Foundation	23.8	23.8	23.8	24.1	24.1
Columns	2	2	2	3	3
Beams	6.5	6.5	6.5	7.2	7.2
Slabs	25.8	25.8	25.8	24.3	24.3

Table (10)
Productivities of each crew formation in cubic meters per day

Crew formation:	1	2	3	4	5	6
Excavation	30	40	45	80	100	140
Foundation	90	120	130	220	0	0
Columns	16	24	32	56	0	0
Beams	10	26	40	55	78	0
Slabs	300	500	550	825	0	0

Table (11)

Labour cost of each crew formation in BHD per day

Crew formation:	1	2	3	4	5	6
Excavation	36	60	63	90	40	45
Foundation	210	300	110	192.5	0	0
Columns	150	192	336	110	0	0
Beams	120	135	200	240	425	0
Slabs	70	100	70	160	0	0

Table (12)

Equipment Cost of each crew formation in BHD per day

Crew formation:	1	2	3	4	5	6
Excavation	30	180	79	360	420	600
Foundation	40	60	65	105	0	0
Columns	40	40	65	125	0	0
Beams	65	40	70	100	80	0
Slabs	40	40	520	760	0	0

One challenge in this problem is the high number of repeating units (villas) and interruption days which results in a very large number of variables. Consequently it becomes more difficult to determine the Pareto front compared to ZDT problem where the maximum number of variables was only 30. To overcome this problem, adjustment is made to find whether there are interruptions when scheduling a sample of 5 units. The 5 units consist of 3 units of type T3M and 2 of type D5 where the ratio 3:2 corresponds to 93: 63. The result of the optimization problem showed that there are no interruption days in all activities and that the optimization searches only for the crew formations variables. In this way the size of the optimum decision making problem becomes manageable for all 156 villas. This optimisation is obviously resulted in interruptions of 0 after all units along all activities as shown in Figure (42) and Table (13). Figure (42) shows the Pareto-front obtained using GA using MAT-LAB toolbar as well as the Pareto-front obtained using MOCS.

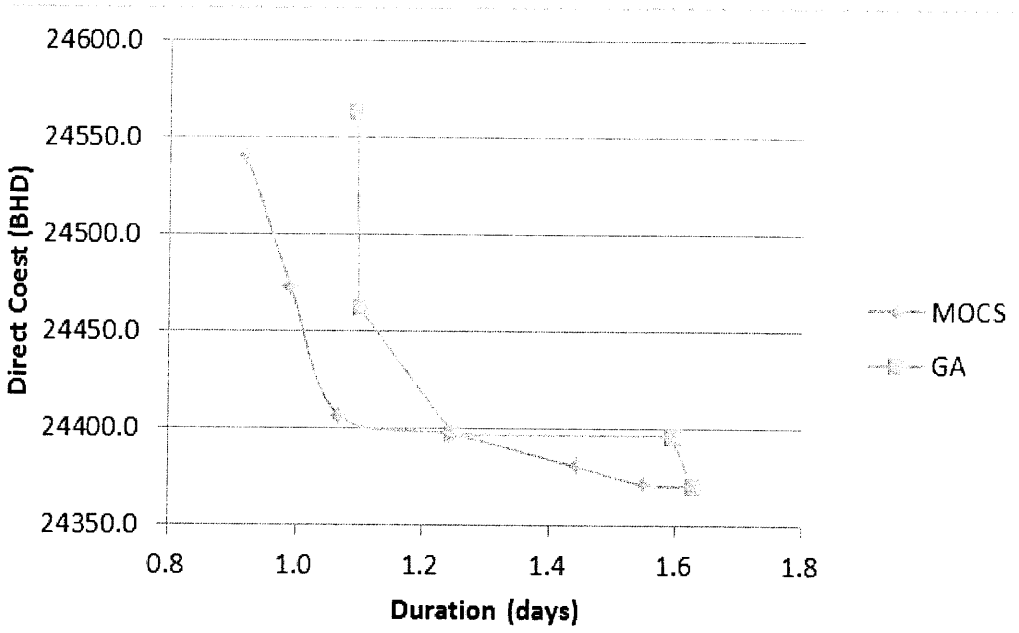


Figure (42)
Pareto front for Case Study 2A of 5 units time-cost trade-off using MOCS and GA

Table (13)
Pareto solutions of Case Study 2A for a sample of 5 villas time and cost Trade-off

Solutions	Project Performance		Selected Crew formation					Total Interruptions
	Project	Project Direct	Excavation	Foundation	Columns	Beams	Slabs	Days
1	0.9	24539.5	4	4	2	5	2	0
2	1.0	24472.8	3	4	2	5	2	0
3	1.1	24407.1	3	4	4	5	2	0
4	1.2	24397.2	3	4	4	4	2	0
5	1.4	24380.6	1	3	4	5	2	0
6	1.5	24371.4	1	4	4	4	2	0
7	1.6	24370.7	1	3	4	4	2	0

Decision Variables

The decision variables are 17 integer variables when only 5 villas are taken in considerations. The first 5 variable are identification numbers (attributes) representing the crew formation while the other variables i.e from 7 to 19 is interruption days. The exact names for each variable are listed as follows:

1. x_1: represents crew formation in activity 1 (Excavation)
2. x_2: represents crew formation in activity 2 (Foundation)
3. x_3: represents crew formation in activity 3 (Column)
4. x_4: represents crew formation in activity 4 (Beam)
5. x_5: represents crew formation in activity 5 (Slab)
6. $inter_{2,1}$: represents interruption days in activity 2 after unit1
7. $inter_{2,2}$: represents interruption days in activity 2 after unit2
8. $inter_{2,3}$: represents interruption days in activity 2 after unit3
9. $inter_{2,4}$: represents interruption days in activity 2 after unit4
10. $inter_{3,1}$: represents interruption days in activity 3 after unit1
11. $inter_{3,2}$: represents interruption days in activity 3 after unit2
12. $inter_{3,3}$: represents interruption days in activity 3 after unit3
13. $inter_{3,4}$: represents interruption days in activity 3 after unit4
14. $inter_{4,1}$: represents interruption days in activity 4 after unit1
15. $inter_{4,2}$: represents interruption days in activity 4 after unit2
16. $inter_{4,3}$: represents interruption days in activity 4 after unit3
17. $inter_{4,4}$: represents interruption days in activity 4 after unit4

Objective Functions

The objectives in the housing project problem considering one contractor are to minimize the direct cost (f_1) needed to perform the five activities and to minimize the duration needed to complete them(f_2) . The objective functions are mathematically defined as follows:

$$\text{Min } f_1 = \sum_{i=1}^{5}\sum_{j=1}^{5}(MC_{i,j} + LC_{i,j} + EC_{i,j}) + IRC \qquad \text{Equation 34}$$

$$\text{Min } f_2 = F_{5,5} - S_{1,1} \qquad \text{Equation 35}$$

Each Activity Duration

The duration is calculated by dividing the quantities over productivities in the same way as were done before (Case Study 1- bridge project). The only difference from the previous formulation is that the parameters are taken from Table (9) to Table (12). It worth pointing out that the model is also not changed and the only change is the parameters in Appendix (5).

Non-negative variables (inequality constraints):

Here also, the parameters affect the lower and upper bounds of the decision variables based on the crew formations available and there are no days of interruptions.

$1 \leq x_1 \leq 6$

$1 \leq x_2 \leq 4$

$1 \leq x_3 \leq 4$

$1 \leq x_4 \leq 5$

$1 \leq x_5 \leq 4$

$0 \leq inter_{2,1}$

$0 \leq inter_{2,2}$

.

.

.

$inter_{4,5} \geq 0$

**All variables are Integer*

Results

The result of this optimisation for 156 units shows the Pareto front in Table (14) and Figure (43). The results are showing that the moderate point in the Pareto front is point 26days costing 761,602BHD which uses method 3, 4 4, 5, and 2 for excavations, foundations, columns, beams, and slabs respectively.

Table (14)
Pareto solutions for Case Study 2A for whole units' time-cost trade-off

Solutions	Project Performance		Selected Crew formation (n_i^*)					Total
	Project Duration (days)	Project Direct Cost (BHD)	Excavation (i=1)	Foundation (i=2)	Columns (i=3)	Beams (i=4)	Slabs (i=5)	Days
1	20	763653	3	4	2	5	2	0
2	24	763344	3	4	2	4	2	0
3	26	761602	3	4	4	5	2	0
4	32	761292	3	4	4	4	2	0
5	35	760796	1	4	4	5	2	0
6	37	760774	1	3	4	5	2	0
7	41	760487	1	4	4	4	2	0
8	43	760464	1	3	4	4	2	0

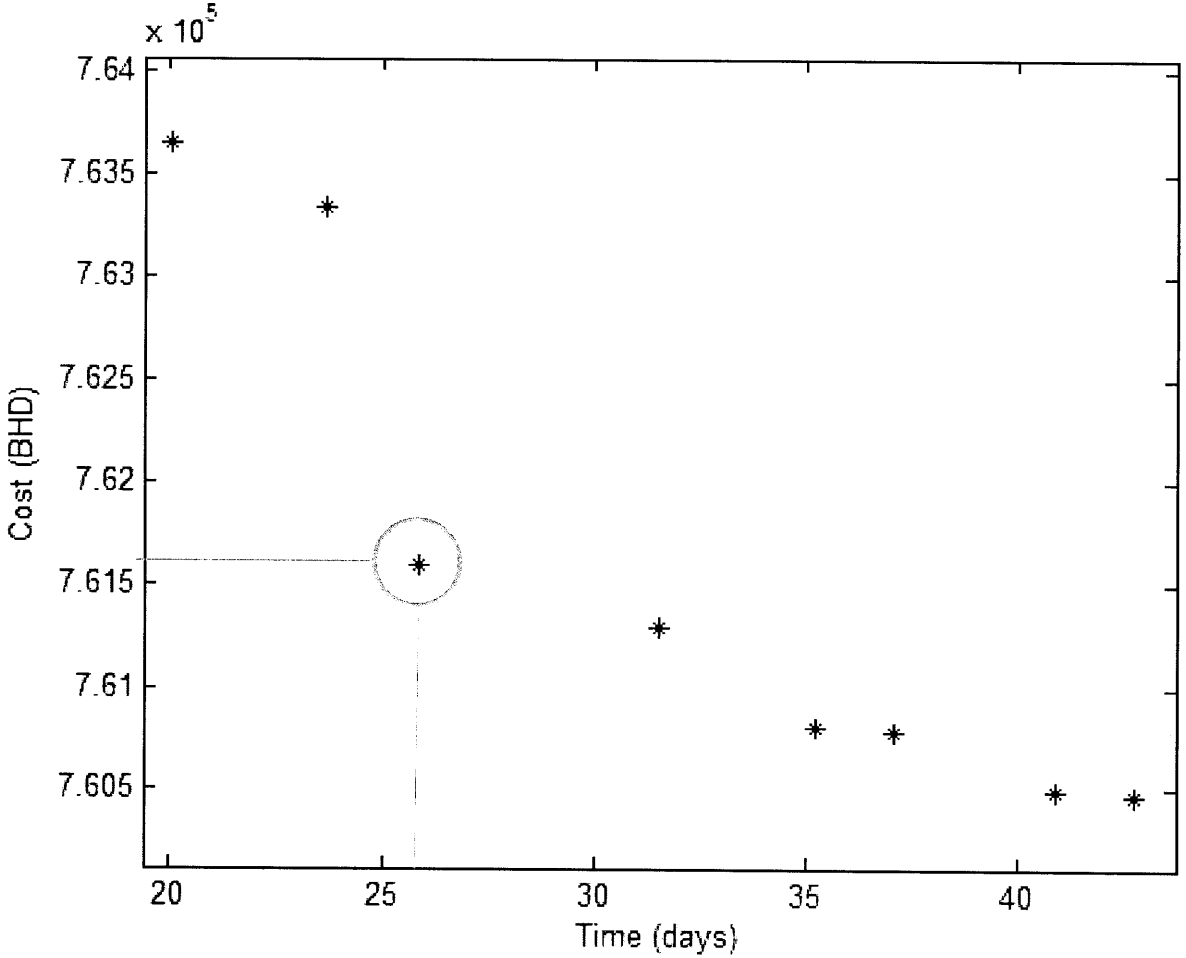


Figure (43)
Pareto solutions Case Study 2A of 156 units' time-cost trade-off

From the data collected about what was actually done in the housing project, Appendix (7) shows the average productivities of the project activities. The cost and time obtained from the productivities and costs in Appendix (3) is 178.8 days and 766,496.4 BHD respectively. These activities are substituted in the cost and time module to find the time and direct cost of the project. The obtained cost and time are used to compare it with the obtained Pareto as shown in Figure (44). This plot clearly shows that the obtained points in the Pareto front are all better than what was done in the actual project in both criteria i.e. time and direct cost.

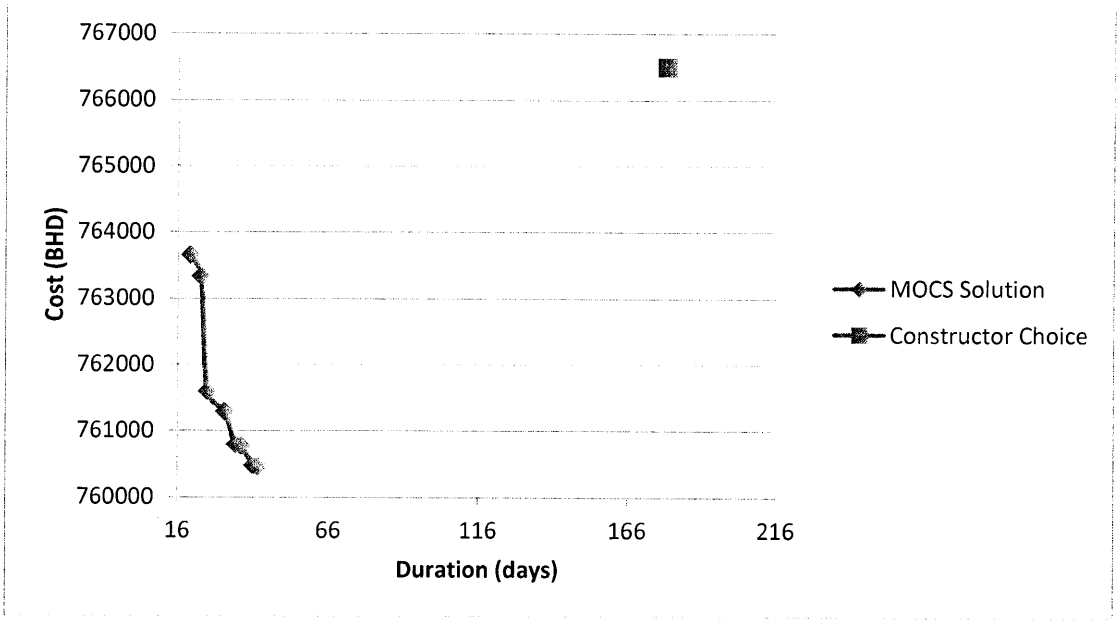


Figure (44)
Comparison between obtained Pareto fronts with the contractor choice

Case Study 2B: Considering various contractors- three contractors

In this section, the optimisation is used to support decision makers to find the best contractor who can be selected for the project and also to select the best crew formations to be used by the contractor. In this case,

the only change from case 2A is that the parameters are divided into matrices each belongs to the referred contractor. This adds one more decision variable, which is the contractor number. That is, there are three contractors considered, contractor1, contractor2, and contractor3 each with different productivities and costs rates as shown in Appendix (3). The optimum decision making problem is extended to determine the most suitable contractor as well. The only difference in the parameters is that they are classified into the contractor who provided the data. Each of the contractors has different crew formations, and direct costs of labour and equipment considering only two methods, normal and fast.

The first variable is identification number of the contractor ($x_{0)}$ and the rest representing the crew formation for the different activities. The interruption days are considered to be zero. The contractors are three as the initial decision variable, then the crew formations decision variables are only two for each contractor and there are no days of interruptions. $1 \leq x_0 \leq 3$

In calculations, the duration is determined by dividing the quantities over productivities in the same way as were done before (case study 1-bridge project). Once again this model ended up having large number of decision variables due to the large number of repeated units. This problem is solved by considering a sample of five villas in order to find out whether there is a need for interruption days or not. The decision variables are 6 integer variables. Table (15) of the optimum solutions shows that the there are no interruption days in the 5 villas, which verifies the reduction of decision variables to include only the crew formations and the contractor number.

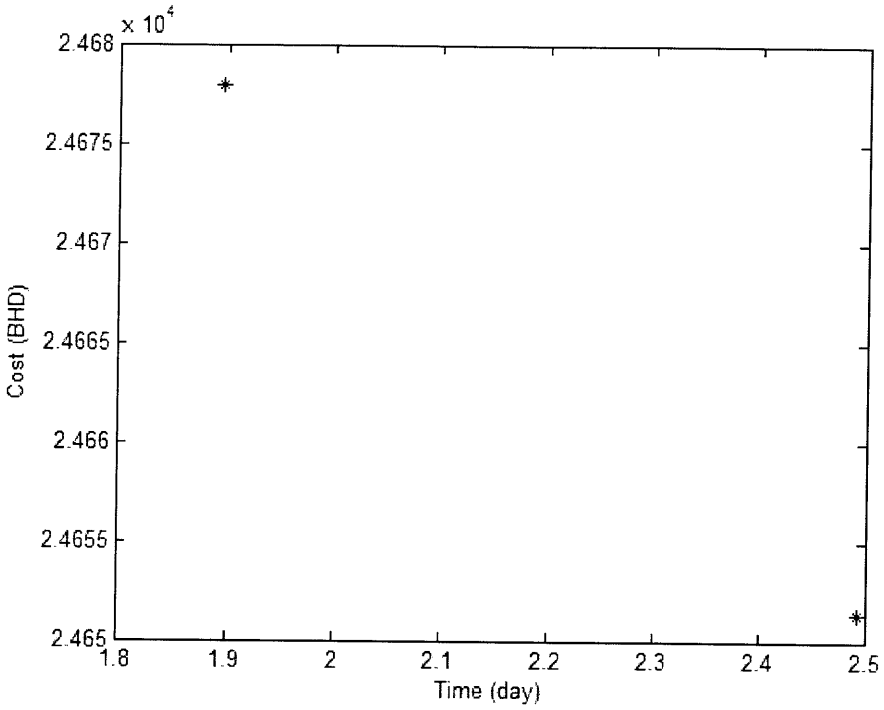


Figure (45)
Pareto optimum for Case Study 2B for 5 villas

Table (15)
Pareto solutions of Case Study 2B for a sample of 5 villas

Solutions	Project Performance		Contractor	Selected Crew formation					Total Interruptions
	Project Duration (days)	Project Direct Cost (BHD)	Number	Excavation (i=1)	Foundation (i=2)	Columns (i=3)	Beams (i=4)	Slabs (i=5)	Days
1	1.9	24677.9	3	1	2	2	2	2	0
2	2.5	24651.4	3	1	1	2	2	2	0

Results

According to the optimization results obtained from Figure (46) and Table (16) , the contractor number obtained from optimization is contractor 3.

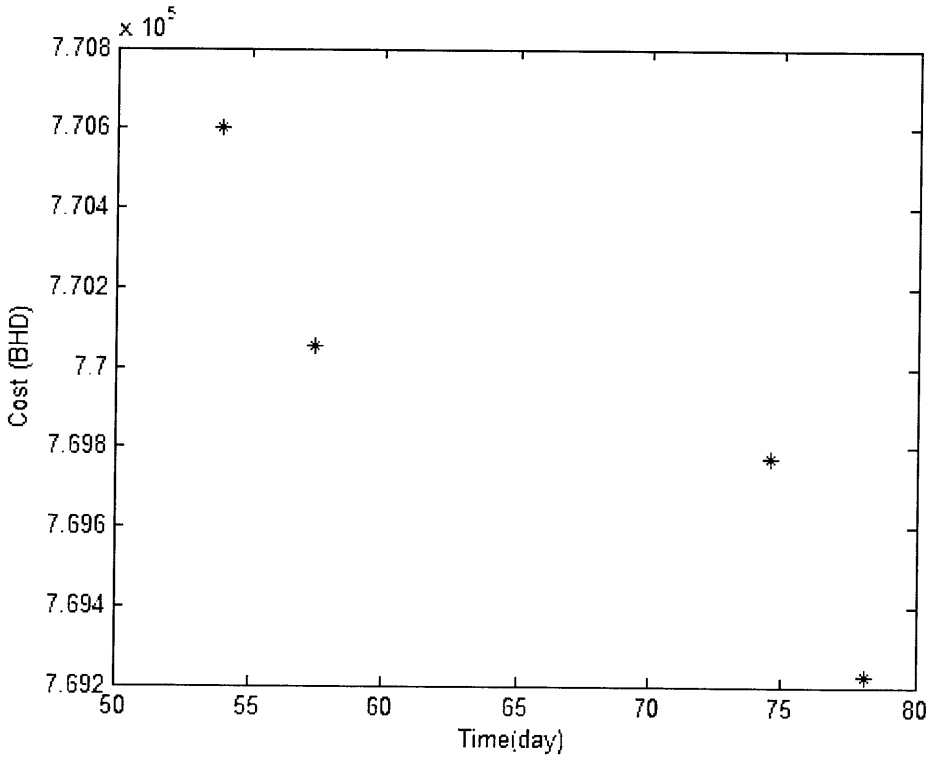


Figure (46)

Pareto front for Case study2B of whole units' time-cost trade-off-- -

Table (16)
Pareto Solutions for Case study 2B for 156 units' time-cost trade-off-

	Project Performance		Contractor	Selected Crew formation (n_i^*)					Total Interruptions
	Project Duration (days)	Project Direct Cost ($)	Number	Excavation (i=1)	Foundation (i=2)	Columns (i=3)	Beams (i=4)	Slabs (i=5)	Days
1	54	770601	3	1	2	1	2	2	0
2	57	770054	3	1	2	2	2	2	0
3	75	769772	3	1	1	1	2	2	0
4	78	769225	3	1	1	2	2	2	0

Chapter 7

Discussion and Conclusions

7.1 Conclusions

7.2 Recommendations

7.3 Future Work

Chapter 7
Discussion and Conclusion

7.1 Conclusion

In this study optimum planning and scheduling of repetitive construction projects is carried out by using MOCS algorithm. Repetitive construction projects such as infrastructure systems and housing projects consist of same activities repeated several times. In such projects it becomes important to arrange the schedule of crew formations such that they have no idle time so that continuation of their work is not interrupted. In repetitive construction projects time and cost are two important coupled factors. Shortening project duration results in increase in the project cost. It is apparent that an optimum balance is required to be determined between these two objectives.

In this study it is shown that repetitive construction projects can be formulated as multi-objective decision making problem and metaheuristic algorithm can be utilized to obtain its solution. In the construction of the mathematical model the project completion time and project cost are taken as objective functions. This turns the optimization problem into multi-objective programming problem. The solution of this optimization problem is obtained by CSA which is one of the recent additions to swarm intelligence techniques. CSA has only two parameters to be initialized before starting the solution which is quite small compared to

other metaheuristic algorithms. These parameters are the total numbers of nests (solutions) and discovering probability p_a.

MOCS algorithm developed in this study is first used to find the Pareto front solution of bridge project taken from (Hyari et. al, 2009) where the Pareto front is obtained by the GA. Comparison of the results has shown that CSA obtains better Pareto front than GA.

Although the values of the Pareto solutions are close and the crew formations are same to those in the previous research, but the interruption days are different. This difference is quite clear in the final decision when the completion incentives, liquidated damages and lane rental cost are included. Particularly, when higher level of information such as liquidated damages and completion incentives are considered, the results obtained became better. That is, when the completion incentives are considered, the minimum duration and costs obtained from MOCS are 108.5days and $1660794 respectively while the resulted values from Hyari et. al 's study are 109 and $1667797 respectively. When considering the liquidated damages the minimum duration and costs obtained from MOCS are 116.3days $1,684,007, respectively, while for Hyari et. al 116.8 days and $1,685,303, respectively. Furthermore, when considering the lane cost the minimum duration and cost are the same in both studies and equals to 123.6 days and 1,792,021.

The application of the MOCS algorithm to housing project in Bahrain has yielded better solution than the one decided by the constructor. The cost and time determined by the constructor are 178.8 days and 766,496.4 BHD respectively. The Pareto front determined by

MOCS algorithm is much smaller than this result. Solutions in the Pareto front range from 20 days to 43 days and the cost range from as 763,653 BHD to 760,464 BHD. The middle value solution in the Pareto front is with duration of 26 days and costs 761,602 BHD. This solution is less than the duration determined from constructor by 85% which costs less by 0.6%. It seems that the decision made by the constructor has not considered the time as an objective but only tried to minimize the cost. There are other solutions in the Pareto front where the project time is less but cost in this case naturally is little more. However the cost is still less than what has been calculated by the constructor. Project manager can select some other solution in the Pareto front depending on whichever she/he prefers.

This study clearly shows that use of optimization techniques in decision making analysis certainly yields better solutions than making decisions intuitively. Furthermore the algorithm developed is also used to select the best contractor for the construction project. Ministry of Housing uses a simple method which is the cost per square meters as criteria for selecting contractors for the construction projects. This may not always yield good solutions. MOCS algorithm is used to select the most suitable constructor for the housing projects among three. In order to keep the size of the decision making problem reasonable only five houses are initially considered in this case instead of 156 houses. This resulted in no interruption values so decision values of the interruption days are not considered when optimising the whole 156 villa. An additional decision variable is considered which represents the contractor number. The problem was solved in both cases when considering one contractor and when considering three contractors. The solution obtained when

considering three contractors of the multi-objective optimization problem has given the most suitable contractor for the housing project as the contractor 3. It is shown that not only the optimum time and cost can be obtained from the model but the most suitable contractor can also be attained.

7.2 Recommendations

This study is best performed when combined with technical information. Specifically; the contractors who aim to adapt new crew formations that help them win the contracts at best price and least time. In addition, the ministry of housing can take the advantage of this study to include comprehensive scopes of works to all possible crew formations. Especially that this resulted Pareto front was far away from what the contractor performed in the project. Moreover, subcontracting can take place when a certain method is needed to control time and cost of the project.

Based on the application of the second part of Case Study 2, it is suggested to use the model in tendering and contractual selection in case of negotiated contracts. For example, when a client like the ministry of housing is looking for an urgent contractor who can build the project at lowest cost, they can perform this study. This part of the study assumes that each of the three contractors has two methods to select from for each activity which is not like the actual life. The ministry of housing can request or keep record of the abilities of numerous methods of various contractors.

7.3 Future Work

Further additions could be made to include reputation or class of contractor and other factors to make tendering a computerized process. In order to apply this study more practically, the data must be very reliable and revised by engineers working for the ministry to ensure good estimation and to keep correct records of what the contractors did or can do. Another proposal is to ask for the methods available for contract when tendering or negotiated contracts wants to take place. In addition, the study needs to be extended to include units in non-series order.

Also, furture works need to adapt the approach of (Zhang et. al, 2013) who obtained better Pareto than the those obtained by Hyari et. al and in this study. Especially that the Pareto obtained by Zhang et. al found better solutions for durations less than 116 days. That is, when considering the interruption days as a voilating constraint, as was done by Zhang et. al, there possibly be a potential improvement in the solutions obtained by MOCS.

References

Samer Ezeldin and Ahmed Soliman (2009). Hybrid Time-Cost Optimization of Nonserial Repetitive Construction Projects. *Journal of Construction Engineering and Management*, 135(1) , 42–55.

Afshar; A. Kasaeian Ziaraty; A. Kaveh; and F. Sharifi (2009). Nondominated Archiving Multicolony Ant Algorithm in Time–Cost Trade-Off Optimization. *Journal of Construction Engineering and Management*, (135), 668–674.

Ahmed B Senouci and Khaled K Naji (2006). A Computerized System for Scheduling and Cost Optimization of Nonserial Linear Projects. *Surveying and Built Environment*, 17(2), 49-62.

Amir Hossein Gandomi, Xin-She Yang, Amir Hossein Alavi (2011). Cuckoo search algorithm: a metaheuristic approach to solve structural optimization problems. *Engineering with Computers*, 29(1), 17-35.

Anon (2014). The Royal Society for the Protection of Birds (RSPB). Retrieved from: https://www.rspb.org.uk/discoverandenjoynature/discoverandlearn/birdguide/name/c/cuckoo/ [Accessed 11 Januaru 2015].

Ashish M Gujarathi1, B V Babu (2009). Improved Strategies of Multi-objective Differential Evolution (MODE) for Multi-objective Optimization.

Deb, K. (2001). Multi-Objective Optimization using Evolutionary Algorithms. *John Wiley & Sons, Ltd*.

Deb, K. (2004). Single and Multi-Objective Optimization Using Evolutionary Computation.

José Rui Figueira, Gabriel Tavares, Margaret M. Wiecek (2009). Labeling algorithms for multiple objective integer knapsack problems. *Computers &Operations Research*, 700-711.

Dréo, J. (2011). Metaheuristics. Retrieved from: http://en.wikipedia.org/wiki/Metaheuristic [Accessed 3 Sep 2014].

Khalied Hesham Hyari , Khaled El-Rayes & Mohammad El-Mashaleh, (2009). Automated trade-off between time and cost in planning repetitive construction projects. Construction Management and Economics,(27), 749–761.

Khalied Hyari and Khaled El-Rayes (2006). Optimal Planning and Scheduling for repetitive. Journal of Management in Engineering, 22(1).

Leonora Bianchi Marco Dorigo Luca Maria Gambardella Walter J. Gutjahr (2008). A survey on metaheuristics for stochastic combinatorial optimization. *Natural Computing,* 8(2), 239-287.

Li-hui Zhang, Xin Zou and Zhi-xiong Su (2013). GA Optimization Model for Time/cost Trade-off Problemin Repetitive Projects Considering Resource Continuity. *Applied Mathematics & Information Sciences*, 7(2), 611-617.

M.P. Saka, E. Dog˘an and Ibrahim Aydogdu, (2013). Analysis of Swarm Intelligence Based Algorithms for Constrained Optimization. *Swarm Intelligence and Bio-Inspired Computation: Theory and Applications,* 25–48.

Mohamed M. Marzouk, Osama A. Omar, Manal S. Abdel Hamid,Moheeb E. El-Said (2010). An optimization algorithm for simulation-based planning of low-income housing projects. *Journal of Advanced Research*, 1, 291–300.

Nakayama, H. (2005). Multi-Objective Optimization and its Engineering Applications.

Rajabioun (2011). Cuckoo Optimization Algorithm Ramin. *Applied Soft Computing, 11,* 5508–5518

Reza Akbaria and Koorush Ziaratia (2010). A multilevel evolutionary algorithm for optimizing numerical functions. *International Journal of Industrial Engineering Computations*, 419–430.

Shu-Shun Liu , Chang-Jung Wang (2007) . Optimization model for resource assignment problems of linear construction projects. *Automation in Construction*, (16), 460–473.

Su-Ling Fan , Kuo-Shun Sun , Yu-Ren Wang (2011). GA optimization model for repetitive projects with soft logic. *Automation in Construction*, (21), 253–261.

T. Warren Liao; P.J. Egbelu; B.R. Sarker; S.S. Leu (2011). . Metaheuristics for project and construction management – A state-of-the-art review. *Automation in Construction*, (20), 491-505.

Talbi, E.-G. (2009). Metaheuristics: From Design to Implementation (Wiley Series on Parallel and Distributed Computing). John Wiley & Sons, Inc.

Tarek Hegazy, and Nagib Wassef, (2001). Cost optimization in projects with repetitive nonserial activities. *Construction engineering and management*, 183-191.

Tohid Erfania; Sergei V. Utyuzhnikova (2010). Directed search domain: a method for even generation of the Pareto frontier in multi-objective optimization'. , *Engineering Optimization*. 467-484.

Xin-She Yang, Suash Deb (2010). Engineering Optimisation by Cuckoo Search. *Mathematical Modelling and Numerical Optimisation*, 1(4), 330–343.

Xin-She Yang, SuashDeb 2011. Multiobjective cuckoo search for design optimization. *Computers & Operations Research*, 40(6), 1616–1624

Xin-She Yang, Suash Deb (2013). Cuckoo search: recent advances and applications. Springer-Verlag London.

Biography

Yang, X.-S. (2010). Engineering Optimization: an Introduction With Metaheuristic Applications. University of cambridge, United Kingdom : A John Wiley & Sons, Inc.

Yang, X.-S. (2010). Lecture: Engineering Optimization and Metaheuristics.

Yang, X.-S. (2010). Nature Inspired Meta-heuristic Algorithms. Second Edition ed. University of Cambridge, United Kingdom: Luniver Press.

Yang, X.-S. (2014). Swarm intelligence based algorithms: a critical analysis. *Springer-Verlag Berlin Heidelberg,* 17-28.

Yuanjie Tang, Rengkui Liu, Quanxin Sun (2014). Schedule control model for linear projects based on linear scheduling method and constraint programming. *Automation in Construction*, (37), 22–37.

Appendices

Appendix (1) MOCS Algorithm Coded Using MATLAB to solve Testing Problems

```
function [Pareto,ParetoSol]=cuckoo_search (n)
if nargin<1,
% Number of nests (or different solutions)
n=50;
end
%Number of eggs per nest(i.e. k=2 for two objectives)
k=2;
% Discovery rate of alien eggs/solutions
pa=0.5;

%% Change this if you want to get better results
% Tolerance (If violation value is applied)
global Tol;
Tol=1.0e-8;
% Number of Iterations
N_iter=5000;  %

%% Simple bounds of the search domain
nd=30; %number of variables for ZDT's (when n=30, much more iterations
are required)
% Lower bounds for SCH
 Lb=-1000*ones(1,nd);

% Upper bounds
% Lower bounds for ZDT's
Lb=zeros(1,nd);

% Upper bounds for SCH
Ub=1000*ones(1,nd);

% Upper bounds for ZDT's
Ub=ones(1,nd);


% Random initial solutions
for i=1:n,
nest(i,:)=(Lb+(Ub-Lb).*rand(size(Lb)));
end

% Get the current best
fitness=zeros(n,1);
[fnew,nest,fitness]=get_best_nest(nest,nest,fitness);

%Starting Values
t=0;
Pareto=[];
ParetoSol=[];
Archfitness=[];

%% Starting iterations

while (t<N_iter),

    % Generate new solutions (but keep the current best)
      new_nest=get_cuckoos(nest,fitness,Lb,Ub);

    %Find wheather the solution is a pareto optimal
        ArchObj=[Pareto; fnew];
```

```
        ArchSol=[ParetoSol; nest];
        A=size(ArchSol,1);
        Archfitness=FitnessFun(ArchObj);
        z=size(ArchObj,1);
        nonDom=ones(z,1);
                        for s = 1:z
                        for so = 1: z

                            if s ~= so

                                        nonDom(s)                                   =
(Archfitness(s)>=(Archfitness(so)))*nonDom(s);

                            end;

                            end;
                        end;
             Pareto = ArchObj(nonDom == 1,:) ;
             ParetoSol =  ArchSol(nonDom == 1,:);
     [fnew,nest,fitness]=get_best_nest(nest,new_nest,fitness);

     % Update the counter
     t=t+n;
     % Discovery and randomization
     new_nest=empty_nests(nest,Lb,Ub,pa);


    % Evaluate this set of solutions
      [fnew,nest,fitness]=get_best_nest(nest,new_nest,fitness);


     % Update the counter again
      t=t+n;


      %Sorting Solution /Ranking based on Fitness

    ArchObj=[Pareto; fnew];

    ArchSol=[ParetoSol; nest];
    A=size(ArchSol,1);
    Archfitness=FitnessFun(ArchObj);

    K=[];
%     A=size(ArchSol,1);
    for i=1:A

        K(i,:)=[Archfitness(i) ArchObj(i,:) ArchSol(i,:)];

    end
    K=sortrows(K,-1 );

%%Finding Current Pareto Points
    N=min(n,A);
    fitness(1:N,:)=K((1:N),1);
    fnew(1:N,:)=K(1:N,2:1+k);
    nest(1:N,:)=K(1:N,4:3+nd);
```

```
            z=size(ArchObj,1);
            for ch=1:z
                for CH=1:z
                    if ch~=CH
                        if ArchObj(ch,:)==ArchObj(CH,:);
                            ArchObj(CH,:)=NaN;
                        end
                    end
                end
            end
            ArchO = [];
            Archf = [];
            ArchS = [] ;
            for s = 1:size(ArchObj,1)
                if ~isnan(ArchObj(s,:))
                    ArchO = [ArchO; ArchObj(s,:)];
                    Archf = [Archf; Archfitness(s)];
                    ArchS = [ArchS; ArchSol(s,:)];
                end
            end
            ArchObj = ArchO;
            Archfitness = Archf;
            ArchSol =  ArchS ;
    z=size(ArchObj,1);
    nonDom=ones(z,1);
                for s = 1:z
                for so = 1: z

                    if s ~= so

                                nonDom(s)                                    =
(Archfitness(s)>=(Archfitness(so)))*nonDom(s);

                    end;

                    end;
                end;
    Pareto = ArchObj(nonDom == 1,:) ;
    ParetoSol =  ArchSol(nonDom == 1,:);


    %Displays during running
    disp(strcat('Iternation Num=',num2str(t)));

    %Current Pareto Error:
    f1 = Pareto(:,1);
    f2 = Pareto(:,2);
%     g=1;
%% for SCH
f3 = (0:100)./100;
f4 = f3.^2;

%% for ZDT1
f3 = (0:100)./100;
  f4 = (1- sqrt (f3));

%% for ZDT2
f3 = (0:100)./100;
f4 = (1-f3).^2;
```

```
%% for ZDT3
f3 = (0:100)./100;
f4 = 1-sqrt(f3).*f3.*sin(10.*pi.*f3);


figure(1);
subplot(1,2,1)
plot(f1,f2,'o',f3,f4)
subplot(1,2,2)
 plot(1:t,e)
end %% End of iterations


%%Posting Results

%% Post-optimization processing
%% Display all the nests
% Pareto;
%          disp(strcat('Total          number          of          Pareto
Points=',num2str(size(Pareto,1))));
% nest
```

Get Cuckoos Get Cuckoo Bird by lévy flights

```
%% Get cuckoos by ramdom walk
function nest=get_cuckoos(nest,fitness,Lb,Ub)
% Levy flights
n=size(nest,1); %number of solutions
% Levy exponent and coefficient
beta=1.5;
sigma=(gamma(1+beta)*sin(pi*beta/2)/(gamma((1+beta)/2)*beta*2^((beta-
1)/2)))^(1/beta);

   B=max(fitness);
   I=nest(fitness==B);
%    r=fix(rand*n)+1;

for j=1:n,
    o=nest(j,:);
    u=randn(size(o))*sigma;
    v=randn(size(o));
% % step length:
s=u./abs(v).^(1/beta);
%
R= fix(rand*size(I,1))+1;
step=abs(s).^(-1-beta);
stepsize=0.1*step.*(o-I(R,:));

    o=o+stepsize;

   % Apply simple bounds/limits
   nest(j,:)=(simplebounds(o,Lb,Ub));
end
```

Discovering and Emptying Worst Nests and Build new Ones in Other Locations

```
%% Replace some nests by constructing new solutions/nests
function new_nest=empty_nests(nest,Lb,Ub,pa)
% A fraction of worse nests are discovered with a probability pa
n=size(nest,1);
% Discovered or not -- a status vector
K=rand(size(nest))>pa;

%% New solution by biased/selective random walks
stepsize=(rand*(nest(randperm(n),:)-nest(randperm(n),:)));
% s=stepsize.*K;
new_nest=(nest+stepsize.*K);
for j=1:n

new_nest(j,:)=simplebounds(new_nest(j,:),Lb,Ub);
end
```

Functions of Testing Problems

```
%%Schaffer's Equations:
function obj=fobj(u)
global Tol;

obj(:,1)=u.^2;
obj(:,2)=(u-2).^2;
```

```
%%ZDT1 Function:
function obj=fobj(u)
global Tol;

d = size(u,2);
for i=1:size(u,1)
x=u(i,:);
g = 1 + 9* sum(x(:,(2:d)),2)/(d - 1);
f1 = x(1);
f2 = g* (1 - sqrt(f1/g));
obj(i,:)=[f1 f2];
end
```

```
%%ZDT2 Function:
function obj=fobj(u)
global Tol;

d = size(u,2);
for i=1:size(u,1)
x=u(i,:);
g = 1 + 9* sum(x(:,(2:d)),2)/(d - 1);
f1 = x(1);
f2 = g*(1 - f1/g)^2;
obj(i,:)=[f1 f2];
end
```

```
function obj=fobj(u)
```

```
global Tol;

%%ZDT3 Function:
d = size(u,2);
%
for i=1:size(u,1)
x=u(i,:);
g = 1 + 9* sum(x(:,(2:d)),2)/(d - 1);
f1 = x(1);
f2 = g.*(1 - (sqrt(f1./g).*f1./g).*sin(10.*pi.*f1));
obj(i,:)=[f1 f2];
end
```

Appendix (2) MOCS Algorithm Coded Using MATLAB to solve Case Studies

```
function
[Pareto,ParetoSol]=cuckoo_search_SCHCOST(n)
if nargin<1,
% Number of nests (or different solutions)
n=50;
end
%Number of eggs per nest (i.e. k=2 for two
objectives)
k=2;
% Discovery rate of alien eggs/solutions
Pa=0.5;
%% Change this if you want to get better results
% Tolerance (If violation value is applied)
global Tol;
Tol=1.0e-8;
% Number of Iterations
N_iter=20000;
 %% Simple bounds of the search domain of Case
Study 1:
Ub=[1 3 3 4 2 10 10 10 10 10 10 10 10 10];
Lb=[1 1 1 1 1 0 0 0 0 0 0 0 0 0];
nd=size(Lb,2);  %number of variables
 % random initial solutions that are rounded to
generate only Integer
for i=1:n,
nest(i,:)=round((Lb+(Ub-Lb).*rand(size(Lb))));
end
 % Get the current best (all solutions are with
zero fitness assuming they are not good)
fitness=zeros(n,1)
[fnew,nest,fitness]=get_best_nest(nest,nest,fitnes
s);
%Starting Values of trials, pareto set, archived
fitness  for the problem
t=0;
Pareto=[];
ParetoSol=[];
Archfitness=[];

%% Starting iterations
while (t<N_iter),
  % Generate new solutions then keep the current
best
```

```
      new_nest=get_cuckoos(nest,fitness,Lb,Ub);

    %Find wheather the solution is a pareto
optimal
        ArchObj=[Pareto; fnew];
        ArchSol=[ParetoSol; nest];
        A=size(ArchSol,1);
        Archfitness=FitnessFun(ArchObj);
        z=size(ArchObj,1);
        nonDom=ones(z,1);
                        for s = 1:z
                        for so = 1: z
                             if s ~= so
                                    nonDom(s)    =
(Archfitness(s)>=(Archfitness(so)))*nonDom(s);

                               end;
                                end;
                        end;
            Pareto = ArchObj(nonDom == 1,:) ;
            ParetoSol =  ArchSol(nonDom == 1,:);

[fnew,nest,fitness]=get_best_nest(nest,new_nest,fi
tness);
     % Update the counter
     t=t+n;
     % Discovery and randomization
     new_nest=empty_nests(nest,Lb,Ub,Pa);
   % Evaluate this set of solutions

[fnew,nest,fitness]=get_best_nest(nest,new_nest,fi
tness);
  % Update the counter again
      t=t+n;

 %Sorting Solution /Ranking based on Fitness
    ArchObj=[Pareto; fnew];
    ArchSol=[ParetoSol; nest];
    A=size(ArchSol,1);
    Archfitness=FitnessFun(ArchObj);
    K=[];
%     A=size(ArchSol,1);
    for i=1:A
```

```
        K(i,:)=[Archfitness(i)         ArchObj(i,:)
ArchSol(i,:)];
    end
    K=sortrows(K,-1 );
 %%Finding Current Pareto Points
    N=min(n,A);
    fitness(1:N,:)=K((1:N),1);
    fnew(1:N,:)=K(1:N,2:1+k);
    nest(1:N,:)=K(1:N,4:3+nd);
  z=size(ArchObj,1);
            for ch=1:z
                for CH=1:z
                    if ch~=CH
                        if
ArchObj(ch,:)==ArchObj(CH,:);
                            ArchObj(CH,:)=NaN;
                        end
                    end
                end
            end
            ArchO = [];
            Archf = [];
            ArchS = [] ;
            for s = 1:size(ArchObj,1)
                if ~isnan(ArchObj(s,:))
                    ArchO = [ArchO; ArchObj(s,:)];
                    Archf         =         [Archf;
Archfitness(s)];
                    ArchS = [ArchS; ArchSol(s,:)];
                end
            end
            ArchObj = ArchO;
            Archfitness = Archf;
            ArchSol =  ArchS ;
    z=size(ArchObj,1);
    nonDom=ones(z,1);
                for s = 1:z
                for so = 1: z

                    if s ~= so

                                nonDom(s)              =
(Archfitness(s)>=(Archfitness(so)))*nonDom(s);
```

```
    end;
                    end;
                end;
    Pareto = ArchObj(nonDom == 1,:) ;
    ParetoSol =  ArchSol(nonDom == 1,:);
      %Displays during running
    disp(strcat('Iteration Num=',num2str(t)));

    %Current Pareto Error:
    f1 = Pareto(:,1);
    f2 = Pareto(:,2);
%Show Old Results to Compare the Pareto Fronts
% Pareto set made by Hyari study:
 f3                                                    =
[107;108;109;110;111;112;113;114;115;116;117;118;1
20;121;124;127;132;133;134;135;137;140;141;143];
    f4                                                    =
[1469361;1465115;1443297;1438619;1433941;1423748;1
419070;1405750;1402353;1398856;1382803;1380055;137
3543;1370795;1358021;1352263;1348405;1344897;13426
47;1339139;1336524;1330766;1323400;1317642]; % for
SCH
 figure(1);
    plot(f1,f2,'*',f3,f4)
    h=legend('MOCS','Hyari');
end %% End of iterations

%%Posting Results
  %% Post-optimization processing
%% Display all the nests
% Pareto;
%    disp(strcat('Total    number    of    Pareto
Points=',num2str(size(Pareto,1))));
% nest
 %%  Calling  Objectives:  The  following  can  be
replaced by the objective functions
function obj=fobj(u)
global Tol;
obj=SchedCost(u);
```

Get Cuckoo Bird by lévy flights

```
%% Get cuckoos by ramdom walk
function nest=get_cuckoos(nest,fitness,Lb,Ub)
% Levy flights
n=size(nest,1);
nd=size(nest,2);
% Lévy coefficient
beta=1.5;
sigma=(gamma(1+beta)*sin(pi*beta/2)/(gamma((1+beta
)/2)*beta*2^((beta-1)/2)))^(1/beta);
%Finding best solutions with highest fitness to
use later in finding new solutions
 B=max(fitness);
I=nest(fitness==B);
%Start a for loop to change every nest:
  for j=1:n,

 o=nest(j,:);
    % Lévy flights in finding step size:
 u=randn(size(o))*sigma;
 v=randn(size(o));
% % step length:
s=u./abs(v).^(1/beta);
%Find a random solutions of the best solutions
R= fix(rand*size(I,1))+1;
 %    stepsize=alpha.*u./(abs(v)).^(1/beta)
step=abs(s).^(-1-beta);
 stepsize=0.1*step.*(o-I(R,:));
%

   % Now the actual random walks or flights
    o=o+stepsize;
   % Apply simple bounds/limits that are rounded
to generate only Integer
   nest(j,:)=round(simplebounds(o,Lb,Ub));
end
```

Application of simple Bounds

```
%% Application of simple constraints
function s=simplebounds(s,Lb,Ub)
```

```
   % Apply the lower bound
   ns_tmp=s;
   I=ns_tmp<=Lb;
   ns_tmp(I)=Lb(I);

   % Apply the upper bounds
   J=ns_tmp>=Ub;
   ns_tmp(J)=Ub(J);
   % Update this new move
   s=ns_tmp;
```

Find the Best Solutions

```
%% Find the current best nest
function
[fnew,nest,fitness]=get_best_nest(nest,newnest,fit
ness)
% this part is made to evaluate all solutions to
find the best
n=size(nest,1);
%One big nest that considers all solutions then
find the dominating solutions only
bignest=[nest;newnest];
bigobj=fobj(bignest);
bigfit=FitnessFun(bigobj);
fitness=bigfit(1:n,1);
fitness1=bigfit((n+1):2*n,1);
for i=1:size(nest,1),
    if fitness1(i)>=fitness(i),
        fitness(i)=fitness1(i);
        nest(i,:)=newnest(i,:);
    end
end
fnew=fobj(nest);
```

Fitness function or Quality of Solution

```
% The larger value the better the solution is in
terms of domination
function f=FitnessFun (obj)
nf = size(obj,2);
SN = size(obj,1);
fDom=zeros(SN,1);
for j = 1:SN
    for jo = 1:SN
        if j~= jo
```

```
            fDom(j)        =        fDom(j)        +
min(obj(j,:)>=(obj(jo,:))).*
~(min(obj(j,:)==(obj(jo,:))==1));
          end
    end
end
fDom = SN - fDom;
f = fDom/SN;
```

Discovering and Emptying Worst Nests and Build new Ones in Other Locations

```
%% Replace some nests by constructing new
solutions/nests
function new_nest=empty_nests(nest,Lb,Ub,Pa)
% A fraction of worse nests are discovered with a
probability Pa
n=size(nest,1);
nd=size(nest,2);
% Discovered1 or not 0 vector
K=rand (size (nest))>Pa;
% New solution by selective random walks and
related to the difference in solutions because
%the cuckoo will discover the alien eggs by the
difference with theirs
stepsize=(rand*(nest(randperm(n),:)-
nest(randperm(n),:)));
% s=stepsize.*K;
%then new solution that are rounded to generate
only Integer
new_nest=round(nest+stepsize.*K);
%Check the New Solutions Satisfies the Simple
Bounds:
for j=1:n
 new_nest(j,:)=simplebounds(new_nest(j,:),Lb,Ub);
end
```

Appendix (3) Data of Three Contractors

(Class A)

Data of three contractors

Contractor Name:	**Alghanah Contracting**							
	Normal Speed			Faster Speed			Explain how briefly	
Activities	Productivity (m^3/day)	Labour Cost (BHD/day)	Equipment Cost (BHD/day)	Productivity (m^3/day)	Labour Cost (BHD/day)	Equipment Cost (BHD/day)	Normal Speed	Faster Speed
Excavation	40m^3/day	60 bhd/day	180 bhd/day	80/day	90 bhd/day	360 bhd/day		
Foundation	16m^3/day	250 bhd/day	40 bhd/day	35m^3/day	600 bhd/day	80 bhd/day		
Columns	6m^3/day	150 bhd/day	40 bhd/day	14m^3/day	450 bhd/day	80 bhd/day		
Beams	26m^3/day	135 bhd/day	40 bhd/day	78m^3/day	425 bhd/day	80 bhd/day		
Slabs precast	550m^2/day	70 bhd/day	520 bhd/day	825m^2/day	160 bhd/day	760 bhd/day		precast

Contractor Name:	**Almoaayed**							
	Normal Speed			Faster Speed			Explain how briefly	
Activities	Productivity (m^3/day)	Labour Cost (BHD/day)	Equipment Cost (BHD/day)	Productivity (m^3/day)	Labour Cost (BHD/day)	Equipment Cost (BHD/day)	Normal Speed	Faster Speed
Excavation	30	36	30	45	63	78.75	8hrs	8+4 hrs
Foundation	130	110	65	220	192.5	105	8hrs	8+4 hrs
Columns	32	192	65	56	336	125	8hrs	8+4 hrs
Beams	10	120	65	20	210	125	8hrs	8+4 hrs
Slabs	25	108	65	50	190	125	8hrs	8+4 hrs

Data of three contractors

Contractor Name:	**Dar Al Khaleej W. L. L**							
	Normal Speed			Faster Speed			Explain how briefly	
Activities	Productivity (m^3/day)	Labour Cost (BHD/day)	Equipment Cost (BHD/day)	Productivity (m^3/day)	Labour Cost (BHD/day)	Equipment Cost (BHD/day)	Normal Speed	Faster Speed
Excavation	**100**	**40**	**420**	**140**	**45**	**600**		
Foundation	**90**	**210**	**40**	**120**	**300**	**60**		
Columns	**16**	**110**	**40**	**24**	**150**	**40**		
Beams	**40**	**200**	**70**	**55**	**240**	**100**		
Slabs (precast slab erection	**300**	**70**	**40**	**500**	**100**	**40**		

Appendix (4) Time and Cost Modules

Scheduling Module

```
%%%%%Duration Calculation:%%%%%

%calculating the duration of the activities based
on the construction
%methods
d=[];
i=0;
j=0;
for i=1:size(Q,1)
   for j=1:size(Q,2)
    d(i,j)=(Q(i,j))./P(i,o(i));
   end
end


%Initializing Finishing times, starting timesm and
logical relations
Fcrew=zeros(size(Q,1),size(Q,2));
Screw=zeros(size(Q,1),size(Q,2));
S=zeros(size(Q,1),size(Q,2));
F=zeros(size(Q,1),size(Q,2));
Slogic=zeros(size(Q,1),size(Q,2));
dalta=zeros(size(Q,1),size(Q,2));
shift=ones(size(Q,1),1) ;


 for i=1:size(Q,1)+1


     for j=1:size(Q,2)+1
            %Finishing time of an acticity should
be after the starting
            %time of the crew adding to the
duration of the activitie
             if i<=size(Q,1) && j<=size(Q,2)
            Fcrew(i,j)=Screw(i,j)+d(i,j);
             end
            %The crew start next unit after the
finishing of the
            %prevous in addition to the
interruption days
            if i<=size(Q,1) && j<=(size(Q,2)-1)
```

```
            Screw(i,j+1)=Fcrew(i,j)+inter(i,j);
            end
            % The logical time to start the e next
activity is after the
            % finish time of the previous activity
            if    i>1    &&    j<=(size(Q,2))    &&
i<=(size(Q,1))
            if Slogic(i,j)<F(i-1,j)+lag(i-1,j);
                Slogic(i,j)=F(i-1,j)+lag(i-1,j);
            end
            end
     end



     for j=1:size(Q,2)+1
        dalta=abs(Slogic-Screw);%between    logical
start and crew start
        dalta(1,:)=0;%No time difference is needed
for the first unit
        shift=(max     (dalta,[],2));     %maximum
difference

           if i<=size(Q,1)
               if j<=size(Q,2)

                    S(i,j)=Screw(i,j)+shift(i);
                    F(i,j)=Fcrew(i,j)+shift(i);


               end
           end
     end


    end



D=F(size(Q,1),size(Q,2))-S(1,1);
```

Cost Module

```
%%%%%Cost Calculations:%%%%


%%Materials Cost (fixed regardless to schedule)
MC=zeros(size(Q,1),size(Q,2));
for j=1:size(Q,2)
for i=1:size(Q,1)
MC(i,j)=M(i,1)*(Q(i,j));
end
end



%% Calculating Cost due to working durations
%Labor Cost
LC=zeros(size(Q,1),size(Q,2));
for j=1:size(Q,2)
for i=1:size(Q,1)

    LC(i,j)=d(i,j)*C(i,o(i));

end
end


%Equipment Cost
EC=zeros(size(Q,1),size(Q,2));
for j=1:size(Q,2)
for i=1:size(Q,1)

    EC(i,j)=d(i,j)*E(i,o(i));

end
end


%%Calculating Interrupting Cost Due to unworking
labour

IRC=0;
```

```
for i=1:size(Q,1)


IRC=IRC+sum(inter(i,:))*C(i,o(i))+sum(inter(i,:))*
E(i,o(i));

end

%%%Cost without Interruptions
DDC=zeros(size(Q,1),size(Q,2));
for j=1:size(Q,2)
for i=1:size(Q,1)

    DDC(i,j)=MC(i,j)+LC(i,j)+EC(i,j);

end
end

%%%Total Direct cost

DC=0;

for i=1:size(Q,1)

    DC=DC+sum(DDC(i,:));


End
DC=DC+IRC;


R=sum(inter);
SS=[];
% f1 and f2 are respectively time and cost:
f1=D;
f2=DC;



SS = [f1 f2];
ObjVal(:,t)= SS;

fViolation(t)=0;
```

```
% Inequality constraints

%   fViolation(i)=sum(g(g>0));   %   no   violations
considered
End %% end of calculations for this row in the
nest
fViolation = fViolation';
```

Appendix (5) Parameters Used in Case Studies

Case Study 1 Parameters

```
function [ObjVal fViolation]= SchedCost(nests)


format bank;
x = nests;

n= size(x,1);
d = size(x,2);

%Initial Violation
fViolation=zeros((size(nests,1)),1);
%Material Cost per Quantity
M=[0; 92; 479; 195; 186];
%Quantity Per Activities(i) per Units(j)
Q=[1147 1434 994 1529; ...
    1032 1077 943 898; .
    104 86 129 100; ...
    85 92 101 80;...
    0 138 114 145];
%Lag Time
lag=zeros(size(Q,1),size(Q,2));
%Productivity Per Activities(i) per Formation o(i)
P= [91.75 0 0 0; ...
    89.77 71.81 53.86 0;...
    5.73 6.88 8.03 0; ...
    9.9 8.49 7.07 5.66;...
    8.73 7.76 0 0];
%Labor Cost per Activities(i) per Formation o(i)
C=[340 0 0 0;...
    3804 2853 1902 0; ...
    1875 2438 3000 0; ...
    3931 3238 2544 1850;
    2230 1878 0 0];
%Equipment Cost per Activities(i) per Formation
o(i)
E=[566 0 0 0; ...
    874 655 436 0; ...
    285 371 456 0; ...
    315 259 204 148; ...
    177 149 0 0];
```

```
for t=1:n %start of calculations for this row in
the nest

u=nests(t,:); %take the decision variables:
%u(1)=Excavation   u(2)= foundation    u(3)= Beams
u(4)= Columns  u(5)=Slabs
%The rest of decision variables are interruption
days


inter(size(Q,1),:)=0;

%locating decision variables in the nests
i=0;
j=0;
for i=2:size(Q,1)-1
   for j=1:size(Q,2)-1
    inter(i,j)=u(size(Q,1)+(size(Q,2)-1)*(i-2)+j);
   end
end

%locating the crew formation number from decision
variables
for i=1:size(Q,1)
    o(i)=u(i);
end
```

Case Study 2 Parameters: when one contractor is considered

```
function [Q ObjVal fViolation]= SchedCostCS(nests)

%%[    Activity1#,    Activity1#,    ..Activityn#,
interruptions...]
format bank;
% nests = round(nests);

x=nests;
n= size(x,1);
d = size(x,2);

z=93; % replace with 3 of T3M for a sample of 5
villas units
zz=63; %replace with 2 of type D5 for a sample of
5 villas
```

```
A=5; %% # of Activities

M=[0.9; 90; 120; 130; 57.3];

q=[5; 23.8; 2;6.5;25.8]; %Type T3M
qq=[6;24.1; 3; 7.2; 24.3]; %Type D5

Q=ones(A,(z+zz));
for f=1:z
Q(:,f)=q;
end
for f=z+1:(zz+z)
Q(:,f)=qq;
end


%Lag Time
lag=zeros(size(Q,1),size(Q,2));
%Productivity Per Activities(i) per Formation o(i)


            P= [30 40 45 80 100 140 ; ...
            90 120 130 220 0 0 ;...
            16 24 32 56 0 0 ;...
            10 26 40 55  78 0;...
            300 500 550 825 0 0];


            C=[36 60 63 90   40 45;
            210 300 110 192.5 0 0; ...
            150 192 336 110 0 0;...
            120 135 200 240  425 0;...
            70 100 70 160 0 0];


            E=[30 180 79 360   420 600;...
            40 60 65 105 0 0 ;...
            40 40 65 125 0 0;...
            65 40 70 100 80 0;...
            40 40 520 760 0 0];
for t=1:n %start of calculations for each row in
the nest
```

```
u=nests(t,:); %take the decision variables:
%u(1)=Excavation  u(2)= foundation  u(3)= Beams
u(4)= Columns  u(5)=Slabs
%the rest of decision variables are interruption
days

%%Start Time and Cost Models as in Appendix (4)
```

Case Study 2 Parameters: when more than one contractor is considered

```
function [ObjVal fViolation]= SchedCostCS (nests)

%% Variables:[contractor#, Activity1#, Activity1#,
..Activityn#, interruptions...]

format bank;
% nests = round(nests);

x=nests;
n= size(x,1);
d = size(x,2);

%Initial Violation
fViolation=zeros((size(nests,1)),1);

z=93;
zz=63;

A=5; %% Activities


M=[0.9; 90; 120; 130; 57.3];

q=[5; 23.8; 2;6.5;25.8];
qq=[6;24.1; 3; 7.2; 24.3];

Q=ones(A,(z+zz));
for f=1:z
Q(:,f)=q;
end
```

```
for f=z+1:(zz+z)
Q(:,f)=qq;
end


%Lag Time
lag=zeros(size(Q,1),size(Q,2));
%Productivity Per Activities (i) per Formation
o(i)
cont=nests(:,1); % a vector of decision variable
of contractor number

% end
for t=1:n %start of calculations for each row in
the nest

u=nests (t,2:((A-2)*(z+zz-1)+6)); %assign the
decision variables
% where u(1)=Excavation u(2)= foundation u(3)=
Beams u(4)= Columns u(5)=Slabs and the rest of
variables are interruptions

%Material Cost per Quantity
switch(cont(t)) %where cont is the vector of the
variable (contractor number)

    case 1 %%Al-Ghinah
    %Productivity of Activities (i) per Formation
o(i)
        P= [40 80 ; ...
            16 35;...
            6 14;...
            26 78;...
            550 825];
    %Labor Cost per Activities(i) per Formation
o(i)
        C=[60 90;
            250 600; ...
            150 450;...
            135 425;...
            70 160];
    %Equipment Cost per Activities(i) per
Formation o(i)
```

```
        E=[180 360;...
            40 80;...
            40 80;...
            40 80;...
            520 760];

            case 2 %%Al-Moayed
    %Productivity of  Activities(i) per Formation
o(i)
        P= [30 45 ; ...
            130 220;...
            32 56;...
            10 20;...
            25 50];
    %Labor  Cost  per  Activities(i)  per  Formation
o(i)
        C=[36 63;
            110 192.5; ...
            192 336;...
            120 210;...
            108 190];
    %Equipment   Cost   per   Activities(i)   per
Formation o(i)
        E=[30 79;...
            65 105;...
            65 125;...
            65 125;;...
            65 125;];

    case 3 %%Dar-Al-Khaleej
    %Productivity of  Activities(i) per Formation
o(i)
        P= [100 140 ; ...
            90 120;...
            16 24;...
            40 55;...
            300 500];
    %Labor  Cost  per  Activities(i)  per  Formation
o(i)
        C=[40 45;
            210 300; ...
            110 150;...
            200 240;...
```

```
            70 100];
    %Equipment   Cost   per   Activities(i)   per
Formation o(i)
        E=[420 600;...
            40 60;...
            40 40;...
            70 100;...
            40 40];

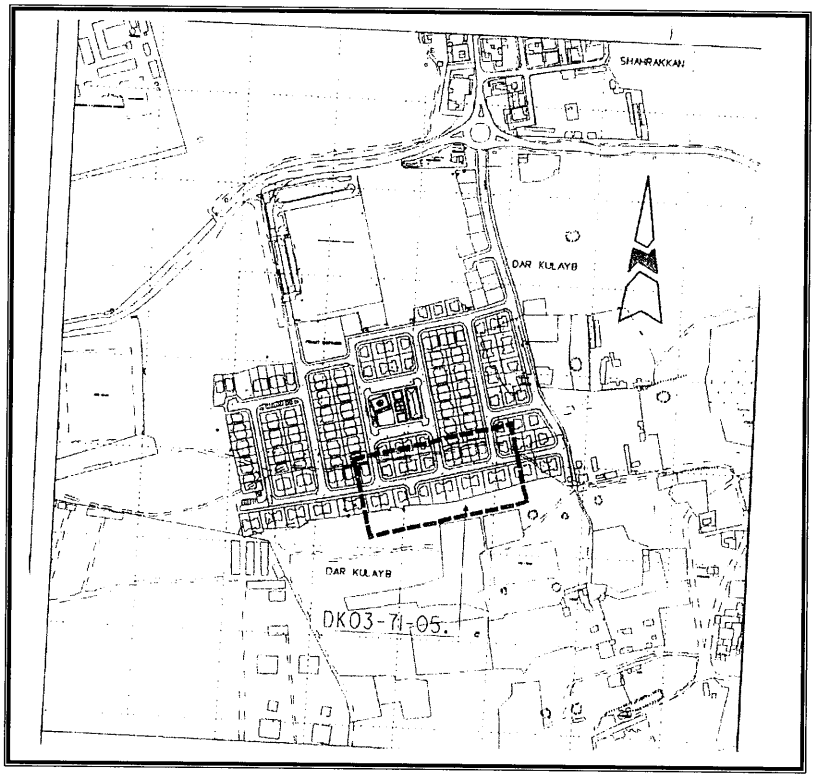
End %end of the switch


%%Start Time and Cost Models as in Appendix (4)
```

Appendix (6) Darkulaib Housing Project

Outline

PROJECT: 156 NOS. TOWNHOUSES TYPE T3M & D5
SITE 145, AT DARKULAIB



GENERAL VIEW

Project Outline

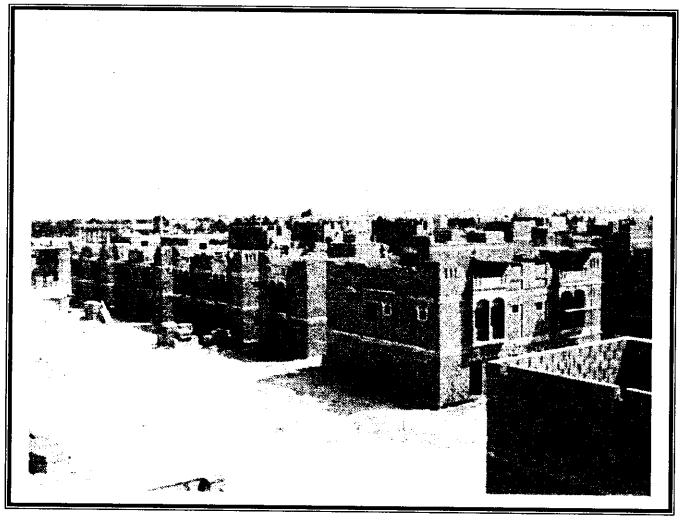
The project consists of construction of 93 nos of Type T3M & 63 nos of Type D5 two storey townhouses with stair cabin located at the area of Darkulaib, site 145, block 1048 including site preparation and earthworks.

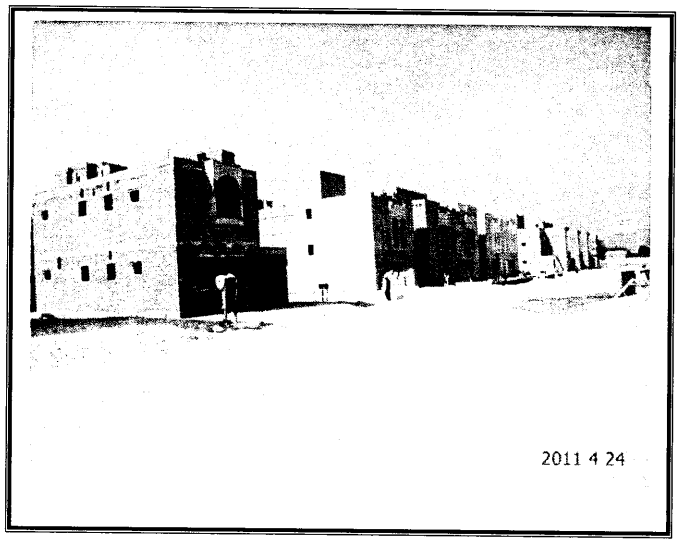
Unit Details

T3M	= 93 units	Area = 208.96m²
D5	= 63 units	Area = 197.76m²
Total	= 156 units	

Overall Contract Areas: 8.0 hectares approx.

PROJECT: 156 NOS. TOWNHOUSES TYPE T3M & D5 SITE 145, AT DARKULAIB





Contract Dates:

Start Date: 24th July 2010

Completion Date: 23rd October 2011

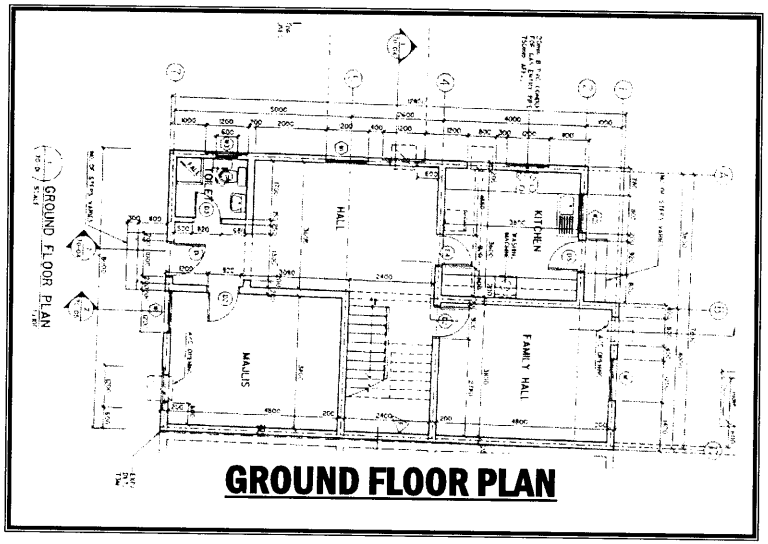
Financial Details

T3M = BD 32,917.360 / units

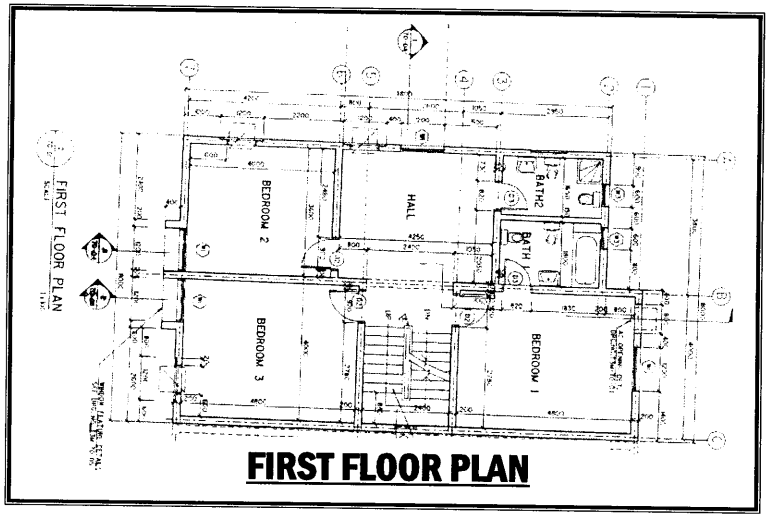
D5 = BD 31,437.030 / units

Overall Cost of Contract = BD 5,041,834.00

PROJECT: 156 NOS. TOWNHOUSES TYPE T3M & D5
SITE 145, AT DARKULAIB



GROUND FLOOR PLAN

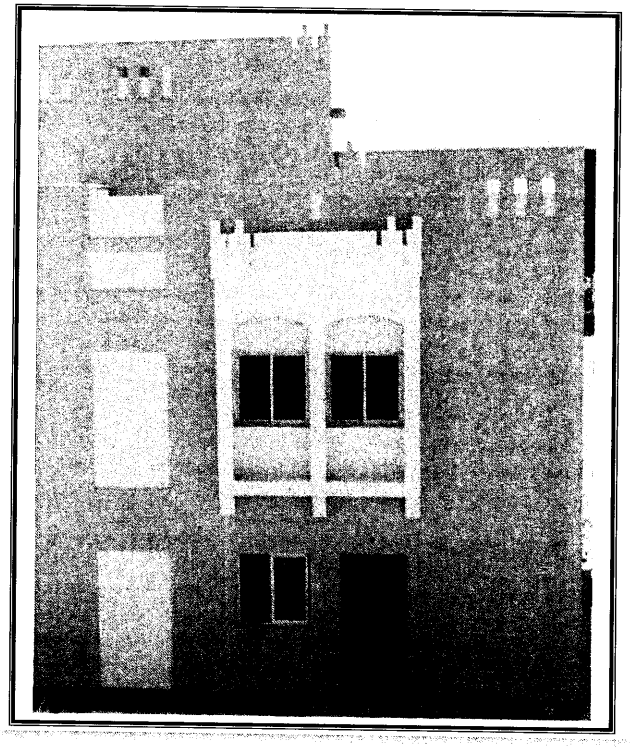


FIRST FLOOR PLAN

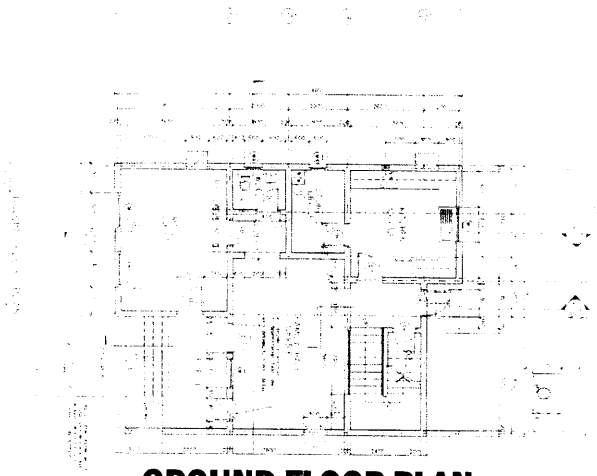
Type: T3M - Double Storey House

Bedrooms	= 3 nos
Halls	= 2 nos
Family Hall	= 1 no
Majlis	= 1 no
Kitchen	= 1 no
Toilet	= 1 no
Bathroom	= 2 nos

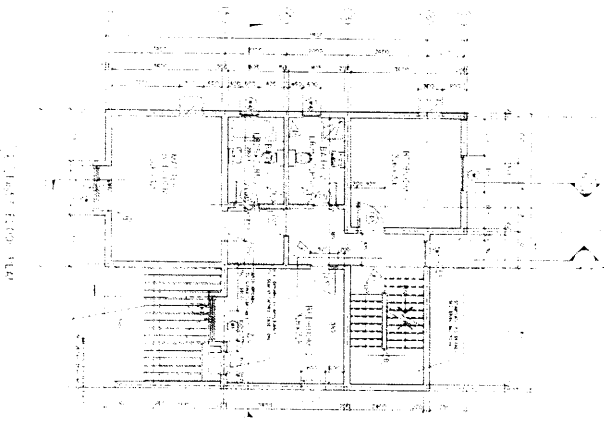
Others:
- ✓ Staircase room
- ✓ Covered Garage
- ✓ Boundary wall



PROJECT: 156 NOS. TOWNHOUSES TYPE T3M & D5
SITE 145, AT DARKULAIB



GROUND FLOOR PLAN

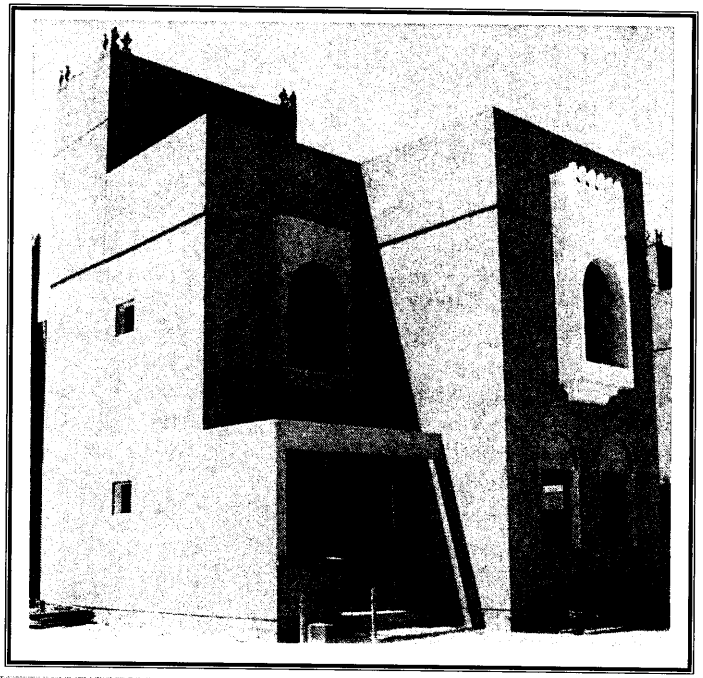


FIRST FLOOR PLAN

Type: D5 – Double Storey House

Bedrooms	= 3 nos
Family Hall	= 1 no
Majlis	= 1 no
Kitchen	= 1 no
Toilet	= 1 no
Bathroom	= 2 nos
Store	= 1 no

Others:
- ✓ Staircase room
- ✓ Covered Garage
- ✓ Boundary wall



Appendix (7) Contractor Data for Darkulaib Housing Project

Activities Productivity and Costs

Name of Project: 156 NOS. TOWNHOUSES TYPE T3M & D5 SITE 145, AT DARKULAIB

Start Date: 24th July 2010

Completion Date: 23rd October 2011

Project duration: 457 Working days

Number of Villas: T3M = 93 units D5= 63 units Total = 156 units

	Villas Type1	Villas Type2
Activities	Quantities m^3	Quantities m^3
1. Excavation	5	6
2. Foundation	23.8	24.1
3. Columns	2	3
4. Ground Beams	0	0
5. Ground Slabs	0	0
6. Columns Extending	0	0
7. Beams	6.5	7.2
8. PC Slab	86 m^2x2=172	81 m^2x2=162
9. Columns Extending	0	0
10. Roof screed	14.4	13.95
11. Finishing (tiles)	43	41

Activity	Possible Crew Formations	Productivity m^3/day	Labor Cost BD/day	Equipment Cost BD/day
1. Excavation	*1 jcb+ 3 laboours+ 1mason*	*5*	28	50
	13 labours + I mason	*5*	*84.4*	*nil*
	8labours+ I mason	*2.5*	*60.8*	*nil*
2. Foundation	4masons+6labors+concrete pump	95.2	38	30
3. Columns	2masons+6labors+concrete pump	12	28	30
7. Ring Beams	4masons+6labors+concrete pump	26	38	30
8. PC Slab	4skilled labors+crane	250m^2/day	20	90
10. Roof screed	4masons+6labors+concrete pump	43.2	38	30
11. Finishing (tiles)	4masons+6labors+concrete mixer	40m^2/day	38	15

ملخص الرسالة

معظم مشاريع البناء مثل أنظمة البنية التحتية لمشاريع الإسكان هي متكررة بطبيعتها حيث يتم اعادة أعمال البناء التي تتكرر في المشروع من وحدة لأخرى أو من منزل لآخر. على سبيل المثال، في طاقم حفر الاساسات بحتاج إلى الانتقال من وحدة سكنية إلى أخرى في المشروع الاسكاني مما يجعل الأمر مهما أن تتم جدولة عملهم بطريقة تضمن عدم تقطعهم عن العمل بحيث تقل الكلف والمدة الزمنية للمشروع. هنالك عدة عوامل مترابطة الهامة تؤثر على اي المشروع من ناحية مدة المشروع وكذلك تكلفة المشروع ومن المعلوم بأنه في الادارة الهندسية نسعى الى تحقيق التوازن الأمثل بين هذين الهدفين. في هذه الدراسة تم كتابة برنامج يقوم باستخدام البيانات كالانتاجية و معدل الكلف من أجل تخطيط وجدولة المشاريع الإنشائية المتكررة بحيث يقلل في نفس الوقت مدة المشروع وتكلفة المشروع. حيث ينتج عن البرنامج مجموعة من الحلول ترسم على شكل منحنى الباريتو الأمثل حيث تتم المفاضلة بين مدة المشروع وتكلفة المشروع.وبعد ذلك سيكون دور مدير المشروع بااختيار أحد الحلول بناءا على متطلبات المشروع. ان البنامج الذي كتب واستخدم يعتمد بالدرجة الأولى على خوارزمية بحث طائر الوقواق حيث اثبت قدرته الفعالة في رسم منحنى الباريتو. ان هذه الدراسة قد أجريت في هذه الأطروحة أيضا على مشروع أسكاني في البحرين حيث تعرض النتائج والاستنتاجات لاحق. إن خوارزمية بحث طائر الوقواق (CS) التي استحدثها شين يانغ وسواش ديب وقد تم تطبيقها بعدإذن في حل مختلف المسائل بشكل فعال. حيث أنها تنتمي إلى خوارميات الأسراب والتي تعتمد في اسلوبها عللا الأنظمة ذاتية للمخلوقات كما هو الحال عند النحل والنمل، وما إلى ذلك من الخوارزميات. وباختصار فإن خوارمية بحث طائر الوقواق (CS) هي خوارزمية مستوحاة من تطفل طائر الوقواق على أعشاش الطيور الأخرى حيث يضع الطائر بيضه في أعشاشهم بطريقة ذكية مظهرا لهم بأنه بيضهم. ومع ذلك، تكتشف بعض الطيور هذه الحيلة وما إن علمت الطيور الحاضنة بذلك حتى تقوم إما برمي البيضة أو ببساطة بهدم العش وبناء آخر في مكان مختلف.

جامعة البحرين

كلية الهندسة

قسم الهندسة الميكانيكية

التخطيط الأمثل وجدولة مشاريع البناء المتكررة بواسطة خوارزمية بحث طائر الوقواق عديدة الأهداف

بحث لاستكمال متطلبات الحصول على درجة (الماجستير) في برنامج الإدارة الهندسية

إعداد

سراء نصير كاظم الأسدي

الرقم الجامعي: 20074441

إشراف

محمد بولات ساكا

(أســـتاذ)

جامعة البحرين

مملكة البحرين

سبتمبر 2015