2009-01-01

Multiple Objective Optimization of Performance Based Logistics

Delia Villanueva
University of Texas at El Paso, dvillanueva3@miners.utep.edu

MULTIPLE OBJECTIVE OPTIMIZATION OF PERFORMANCE BASED LOGISTICS

DELIA VILLANUEVA JAQUEZ

Department of Industrial Engineering

APPROVED:

Heidi Taboada Ph.D., Chair

Eric Smith, Ph.D.

Martine Ceberio, Ph.D.

Patricia D. Witherspoon, Ph.D.
Dean of the Graduate School

For my family, fiancé and friends…

for their love and support throughout my life.

MULTIPLE OBJECTIVE OPTIMIZATION OF PERFORMANCE BASED LOGISTICS

By

DELIA VILLANUEVA JAQUEZ

THESIS

Presented to the Faculty of Graduate School of

The University of Texas at El Paso

in Partial Fulfillment

of the Requirements

for the Degree of

MASTER IN SCIENCE

Department of Industrial Engineering

THE UNIVERSITY OF TEXAS AT EL PASO

December 2009

ACKNOWLEDMENTS

"Thank God because you gave me the strength to make the decision to initiate this adventure and the strength to complete it"

I would like to thank all the people who have helped and inspired me during my Graduate study. My thesis advisor, Dr. Heidi Taboada, who was always there and helped me make this work a success. Her encouragement and guidance have enabled me to complete this thesis. Also, thanks to Dr. Espiritu; his assistance and support helped me to achieve this important goal in my professional life.

I am grateful to my mom and dad, from whom I have received so much love and guidance. For their confidence and support, it would have been simply impossible without them. The constant love and support of my sister is sincerely acknowledged. Thanks for being always by my side.

Love and friendship are essential to overcome difficult times and maintain hope in the future. Special thanks to my Fiancé who has been a light in my life. Along the way, there were many times in which his love and company were essential to continue. To my lab friends at the Ergonomics, Safety and Productivity Applications Laboratory, who know better than anyone about the difficulties of graduate student life. They were always there to answer any of my questions and to get their comments on any new ideas. All this time would not have been so enjoyable without their company.

Last but not least, thanks to the Chihuahua State Government for the financial support along my master's.

ABSTRACT

"Multiple Objective Optimization of Performance Based Logistics"

By: Delia Villanueva

Thesis Director: Dr. Heidi A. Taboada

This thesis presents a new Performance Based Logistics optimization model. Performance Based Logistics (PBL) is becoming increasingly important for manufacturers in mission critical environments that need to provide ultimate product availability at the lowest cost and with the highest level of customer satisfaction. The U.S. Department of Defense has mandated that Performance Based Logistics programs be adopted by its major weapon systems and equipment suppliers, is one of the newest support strategies to improve the weapon system readiness. This work presents a new multiple objective evolutionary approach that simultaneously optimizes objectives such as Reliability, Maintainability and Total Cost for Ownership. These objectives are based on procurement, operation and maintenance components of the Total cost of ownership.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1. INTRODUCTION

This thesis is focused on the development of a new multiple objective optimization method for Performance Based Logistics (PBL). Performance Based Logistics is becoming increasingly important for manufacturers in mission critical environments who need to provide ultimate product availability at the lowest cost and with the highest level of customer satisfaction.

1.1 Performance Based Logistics

The U.S. Department of Defense (DoD) has adopted a new strategy, Performance Based Logistics. Simply stated, on the cost of operations and support every year the U.S Department of Defense expends a lot of money of their available budget, in fact two-thirds of all the defense expenditures (Goure, 2009). In the past, the DoD has dictated to contractors what to produce, when to produce it and the activities that they should carry out. The more a contractor produced, the more money they made (Vitasek & Geary, 2008). However, in 2001 the DoD adopted a new approach in order to reduce operations and support for their weapon systems. PBL focuses on performance outcomes, not in individual parts or repair actions (Vitasek & Geary, 2008; Goure 2009). The main purpose of the DoD is to obtain products/systems that satisfy their needs with measurable improvements and be able to deal with their mission capability and operational support in a timely manner and at a reasonable price.

According with the DoD, “PBL represents the purchase of support as an integrated, affordable, performance package designed to optimized system readiness and meet

performance goals for a weapons system through long-term support arrangements with clear lines of authority and responsibility." (Berkowitz *et al*., 2005; Vitasek & Geary, 2008; Gansler & Lucyshyn, 2006). A good administration is key to significantly impact the national security by moving aggressively to the Performance Based Logistics concept by the simultaneously optimization for all the conflicting objectives.

The main objectives that are considered in this thesis are the maximization of the system reliability, the minimization of maintainability and the minimization of total cost of ownership (TCO). The reliability is the probability that a component or system will perform a required function for a given period of time when used under stated conditions. On the other hand, maintainability is the probability that a failed component or system will be restored or repaired to a specified condition within a period of time when maintenance is performed in accordance with prescribed procedures (Ebeling, 1997). Finally, the cost of ownership serves as the common denominator for a person to understand the technical and nontechnical concepts.

While the speed, range, firepower, and mission performance of weapons/equipment system has improved significantly, the main goal remains the same: achieving specified levels of reliability and maintainability at a lower cost. Crucial objectives for system safety and mission success. Poor reliability or false failure indications for a component may affect directly to the user of such component and can result in the loss of a life. A poor maintainability and unavailable equipment directly affects the mission success and may cause the repetition of the mission.

1.2 Multiple Objective Optimization Problems

Multiple Objective Optimization has been under study research in several different research communities, and although many complex problems have been solved. To solve this kind of problems there are special approaches which can divide in two different techniques: the mathematical approaches and the metaheuristic approaches.

The mathematical approaches are the most traditional and common methods for solving multiobjective optimization problems. These methods aggregate all the different objective functions into a single objective function. The weighed sum method, goal programming and utility theory are good examples for this kind of approach.

In the past, Nowick *et al.* (2007) optimized the reliability, maintainability and supportability objectives under PBL using a goal programming model by grouping all the objectives in a single equation and trying to minimize the total penalties for the undesirable deviations from the specified targets. Calabria *et al.* (1995) used an analytical approach which evaluates and determines the reliability and maintainability allocation which maximizes a given product index or minimizing the total cost of the investments. Unfortunately, for these kinds of methods every objective needs a weight that scalarize its importance and their context in the problem and a target that specified the aspiration levels. They do not attempt to minimize or maximize the objective function, rather than this it seeks to minimize the deviations from the desired goals according to the priorities assigned. However, the difficulty of these types of methods lies in the fact that the weight needs to be assigned by a specialist with expert knowledge in the process so the result could be close to the desired point.

In real world, conflicting objectives come in different scenarios, some of them come by itself, but sometimes there are a group of them, and they cannot be improved without the deterioration of the others. In this case, a tradeoff with the different conflicting objectives is needed in order to find the best possible combination that simultaneously optimizes all the objectives. In recent years the metaheuristic approaches such as Tabu Search, Ant Colony, Particle Swarm, Simulated Annealing and Genetic Algorithms have been widely used with successful applications in different areas, such as: optimization in scheduling, facility layout, supply chain management, maintenance policy selection, spare parts inventory, kanban systems and assembly line planning, among others.

For the optimization of the PBL concept, a multiple objective genetic algorithm seems to be the most suitable option due to all the advantages that genetic algorithms offer. Genetic Algorithms (GAs) are a bio-inspired approach, another class of optimization method which attempts to solve problems guided by a genetic process. This algorithm is inspired in the evolution theory of Charles Darwin. Genetic Algorithms are nondeterministic stochastic search/optimization methods that simulate the process of natural evolution to solve problems with a complex solution space. They are computer-based algorithms that mimic some of the known mechanisms in evolution, as key elements in their design and implementation. This algorithm is able to deal with the information generated in previous iterations or stages.

1.3 Research Objectives

The development of a Genetic Algorithm is proposed in order to optimize several objective functions of importance in the Performance Based Logistics area. There are

many problems in which Genetic Algorithms have been successfully applied. For instance, Coit & Smith (1996) developed a GA to analyze series-parallel systems by determining the optimal design configuration with some system-level constrains on reliability, cost and weight. Paiton & Campbell (1995) presented an optimization model that identified the types of component improvements and level of effort spent on those improvements to maximize the system reliability subject to cost constrains, using GAs. Cheng & Li (1997) presented a constrained multiobjective optimization methodology by integrating a Pareto genetic algorithm and a fuzzy penalty function method. Taboada *et al.* (2008) developed a custom Genetic Algorithm to solve multiple objective multi-state reliability optimization problems. Later, Taboada & Coit (2009) proposed a multiple objective evolutionary algorithm for solving system design allocation problems by developing another type of crossover in which multi-parent recombination is allowed.

The main objective of this research thesis lies on the development of an algorithm that optimizes different objectives under the PBL concept.

1.4 Thesis Organization

In this thesis, multiple objective optimization is implemented using Genetic Algorithms to simultaneously optimize the following objective functions, (*i*) maximization of reliability, (*ii*) minimization of maintenance time and (*iii*) minimization of total cost of ownership.

This thesis is organized as follows: In Chapter 2, a description of the basic principles of multiple objective optimization problems is described, and the summarization of some of the most popular mathematical methods such as the

weighted sum, goal programming and utility theory is presented. In addition, some of the metaheuristic approaches such as: tabu search, simulated annealing, ant colony, particle swarm and genetic algorithm are introduced. In Chapter 3, a description of the proposed Multiple Objective Genetic Algorithms is shown as well as its basic concepts, the general solutions techniques, and all the different genetic operators are presented.

Genetic Algorithms offer different characteristics than offered by other heuristic methods. For instance, one of the most important differences is that a GA works with a population of possible solutions, while other heuristic method uses just one solution in their iterations. Another difference is that GA is probabilistic, not deterministic. Chapter 4 describes the Performance Based Logistics in detail. Performance Based Logistics as the new DoD preferred method of support has gain popularity lately. This concept basically moves the focus from management of parts and supplies to management of the suppliers responsible for delivering the required performance, such as availability and response time. The real meaning of PBL is the purchase of weapons system sustainment as an integrated, affordable package based on output measures such as the weapon system availability, rather than input measures such as parts and technical services.

Chapter 5 presents a simple example in which the goal is to optimize the design of a series system considering conflicting objective functions. Finally, in Chapter 6 a more complex example is used to illustrate the performance of the developed Multiple Objective Genetic Algorithm considering several objective functions characteristic of Performance Based Logistics.

CHAPTER 2. MULTIPLE OBJECTIVE OPTIMIZATION PROBLEMS

2.1 Introduction

Several problems involve the optimization of one single objective function. However, most real world problems involve the optimization of several, usually conflicting objectives. The multiobjective optimization field is very complex and mathematically intensive. Usually, in single objective optimization the search space is well defined. As soon as there are several conflictive objectives to be optimized there is no longer a single optimal solution but rather a whole set of possible solutions (Abraham *et al.*, 2005). Multiobjective optimization involves the simultaneous optimization of two or more conflicting objectives functions.

The mathematical formulation of a multiple objective optimization problem is as follows:

Minimize/Maximize $f_m(x)$,		$m = 1, 2, \ldots, n$
Subject to:	$g_j(x) \geq 0$	$j = 1, 2, \ldots, J$
	$h_k(x) = 0$	$k = 1, 2, \ldots, K$
	$x_i^{(L)} \leq x_i \leq x_i^{(U)}$	$i = 1, 2, \ldots, n$

Where,

$f_m(x) = (f_1(x), \ldots, f_n(x))$ for i=1, 2, …,n

$g_j(x)$=j^{th} inequality constraint evaluated at x.

$k_i(x)$=i^{th} equality constraint evaluated at x.

$f_i(x)$= i^{th} objective function evaluated at x.

$x = \{x_1, \ldots, x_p\}$ is a vector of decision variables.

n= number of objectives or criteria to be optimized.

p= number of decision variables.

When dealing with multiple objectives the solutions can be easily shown in the Pareto Front, where each point represents a good solution for the different objectives in conflict, there is not point better than the others all of them are known as nondominated points. They possibly will not have one solution which is best with respect the others, but they are considered greater than the others in the search space (Taboada & Coit, 2007).

A solution x_1 dominates a solution x_2 if and only if the two following conditions are true:

- x_1 is no worse than x_2 in all objectives, i.e., $f_i(x_1) \leq f_i(x_2)$ for all *i, i* $\in$ {1, 2, …, *n*}.
- x_1 is strictly better than x_2 in at least one objective, i.e., $f_i(x_1) < f_i(x_2)$ for at least one *i.*

One of the most important things is that usually there is no single solution to a problem with multiple objectives. The solution for a multiple objective problem always can be found in its Pareto optimal set. The Pareto optimality concept shows a set of nondominated solutions, which means that no objective can be improved without the degradation at least one of the others objectives. Any point in the Pareto front is considered a Pareto-optimal solution (Martinez *et al.*, 2009).

The multiobjective optimization field is very complex and mathematically complicated for most of the people. In single objective usually the search space is well defined. As soon as there are several contradicting objectives to be optimized there is no longer a

single optimal solution but rather a whole set of possible solutions, and to obtain the optimal solution there will be a set of optimal trade-offs between the conflicting objectives (Abraham *et al*., 2005). For a long time it has been under research in so many different areas and there is a large process to go and find the adequate technique to understand all issues. Multiple objective optimization has a lot of areas under research and there is a considerable way to go still before having an adequate technical understanding of all issues. As the knowledge of complex systems in all areas is increased it is easy to see the need to understand how the objectives are related.

If all objective functions and constrains are linear, the result for the multiobjective optimization problem will be linear as well. But if any of the objective functions or constrains are non linear the result will be a nonlinear multiobjective problem. In general, multiple objective optimization is divided in two main stages: (*i*) the multiple objective optimization part, and (*ii*) the multicriteria decision analysis stage. The main focus of this thesis is on the multiple objective optimization part. There are two general approaches to solve multiple objective optimization problems: the mathematical approaches and the metaheuristic approaches. The differences between the two methods are described next.

2.2 Mathematical Methods

These are the most traditional and common methods for solving the multiobjective optimization problem. They are mainly distinguished from the evolutionary methods because these methods need to aggregate or scalarize all the different objectives into a single objective function.

2.2.1 The Weighted sum method

The weighted sum method scalarizes a set of objectives into a single objective by multiplying each objective by its relative weight. This method is the simplest approach and is probably the most commonly used because its simplicity (Kalyanmoy, 2001). When dealing with multiple objectives this method is the first that comes to mind; the idea is very simple, but is hard to define the weight for each objective. The assignation for each weight would be depending on the importance for the different objectives and their context in the problem.

The weighted sum method for multiobjective optimization is one of the most widely-used methods (Kim & Weck, 2004). This method is able to transform multiple objectives into a single cumulative objective function by multiplying each objective function by a weighting factor and summing up all weighted objective functions as follows:

$$Z = w_1x_1 + w_2x_2 + \ldots + w_mx_m$$

Where w_i ($i = 1, 2, \ldots, m$) is a weighting factor for the i^{th} objective function, such as $0 \leq w_i \leq 1$ the weighted sum is said to be a convex combination of objectives. This kind of method is usually used in cases when the varying significance of the individual criteria is known or can be estimated. An optimal solution for this problem is an efficient point for the original multiobjective model (Kalyanmoy, 2001).

The weight of an objective is usually chosen in proportion to the objective's relative importance in the problem. Nowadays, there exists ways to quantify the weights from this qualitative information because this method requires a precise value of weight for all the different objectives. However, to set up an appropriate weight also depends on the

scaling (Kim & Weck, 2004). It is likely that the different objectives take different orders of magnitude. When the objectives are weighted to form the objective function, it would be better to scale them in a way that each has more or less the same magnitude, this is called normalization process of the objectives (Kalyanmoy, 2001).

2.2.2 Goal Programming

The basic concept of Goal Programming (GP) was initially established by Charnes in 1955, but then it was revised in 1961 by Charnes and Cooper. This is one of the first methods for solving multiobjective optimization problems (Schniederjans, 1995). The main idea in goal programming is that the decision maker specifies aspiration levels for the objective functions and the deviation from these aspirations are the ones that are going to be minimized. The goal is formed by the objective function and the aspiration levels. Goals could be also represented as equalities or ranges; the aspirations levels are assumed to be selected so that they are not achieved simultaneously (Miettinen, 1998).

GP considers multiple goals that are often in conflict with each other. With multiple goals, all goals usually cannot be realized exactly. An example of multiple conflicting objectives can be found in organizations that want to simultaneously: (1) Minimize total cost of ownership; (2) Maximize the reliability of the system; (3) Minimize the mean time to repair; and (4) Minimize the logistic footprint of the system.

The general formulation does not attempt to maximize or minimize a single objective function as the linear programming model does. Rather than this, it seeks to minimize the deviations among the desired goals and the actual results according to the priorities

assigned. The objective function of a goal programming model is expressed in terms of the deviations from the target goals (Nowicki *et al.,* 2007).

It is expressed as follows:

$$Min\ Z = (\omega_1\ |f(x_1) - g_1|) + (\omega_2\ |f(x_2) - g_2|) + \ldots + (\omega_n\ |f(x_n) - g_n|)$$

Where:

ω_n is a non negative constant that represent the relative weight to be assigned to the deviation variables, while $|f(x_n) - g_n|$ represents the deviation variables.

Goal programming was suggested to be used when solving unsolvable Linear Programming (LP) problems, such as the infeasible problems. Many references in Operational Research and Management Science consider GP as one of most mathematical or logic methodologies that exist in the field (Schniederjans, 1995).

2.2.3 Utility theory

The utility theory suggests a practical structure of the evaluation of different alternatives or choices made by individuals, firms and associations. Utility can be translated into the satisfaction that each choice provides to the decision maker. In consequence, utility theory assumes that any decision is made on the basis of the utility maximization principle, all this should be done according to which the best choice that is the one that provides the highest utility or benefit to the decision maker.

The utility theory was firstly presented by B. Pascal and D. Bernoulli in the XVIIIth century. But it was until the Second World War it gained attention because of the study

“How customer best decide?”, an information that mention how to achieve importance in all markets. Before that, it was only used basically to understand the decisions of subjects in scenarios of hazard games (Duarte, 2001).

Utility theory is an area of decision analysis that is considered with how to build models to explain and guide choice behavior under uncertain situations. The main point is to analyze how people do and how they should take decisions in the presence of some kind of risk factor. In this case, the customer plays an important role of the decision maker that must decide which of the many different goods and services to consume so as to secure the highest possible level of total utility subject to his/her available income and the prices of the goods/services (Figueira *et al*., 2005).

Basic axioms

The following are the basic axioms of the modern multiattribute utility theory that were developed by Von Neumann and Morgenstern (Figueira *et al*., 2005; Duarte 2001). They specify conditions on an individual’s preference over pairs of risk prospects. These axioms and they are described as follows:

• *Preference order axiom.*

The subject is able to compare and rank pairs of alternatives. As a consequence there exists a transitive and reflexive function (> preferred to, or ≥ preferred or indifferent to). This helps by ranking the different alternatives.

• *Continuity axiom.*

For the three different alternatives *p, q* and *r*, if *r* is preferred to *q* and *q* is preferred to *p* ($r > p > q$) there exist a real λ for each $\lambda r + (1 - \lambda)\ p \approx q$, where ≈ is the indifferent operation.

• *Independence Preference axiom.*

For three different alternatives *p, q* and *r* and a real λ, if $p > q$ then every convex combination $\lambda p + (1\text{-}\lambda)\ r$ is preferred to every convex combination $\lambda q + (1 - \lambda)\ r$.

These axioms enable to state that if a subject prefers alternative *p* to *q* then the utility for *p* is greater than that of *q*, that is $u(p) > u(q)$.

In real decisions scenarios, the problems are characterized by some attributes and for that reason they reach an overall utility model in a demand task.

One of the most important things in utility theory is the idea or perception of independence of the variables or additivity of values. Its importance stems from numerous multiple-criteria procedures used for rating people, products, and other things. The methodology that is used for this kind of problems is to construct the one dimension utility function and make a combination of them as a multiattribute utility model by evaluating the trade-offs between the different attributes. It is only possible to go from one-dimension utility to the multi-utility function if the independence conditions of decision makers with respect to the attributes are guaranteed. The validation of such conditions assures that they are able to compare between pair alternatives and rank them (Keeney & Raiffa, 1993).

Let's suppose that n consequences are given to start $x_1, x_2, \ldots, x_n$. Each x could be a scalar, a vector or a paragraph. It is important to the decision maker to rank the consequences in order of his preference. Let's assume that x_1 is less prefered that x_2, which is less prefered than x_3, and so on. In other terms: $x_1 < x_2 < x_3 < \ldots < x_n$.

Then, the decision maker is asked to express his preferences for probability distributions over these consequences:

1. His first option will result in consequence x_i with probability p'_i, for $i = 1, 2, \ldots, n$. $p'_i \geq 0$, all i, and $\Sigma_i\, p'_i = 1$.

2. The next option will result in consequence x_i with probability p''_i, for $i=1, 2, \ldots, n$. $p''_i \geq 0$, all i, and $\Sigma_i\, p''_i = 1$.

It is possible to have an infinitive of potential probability distributions over this finite set of consequences.

But then suppose that the decision maker says that, for each i is indifferent between these two options:

1. Certain option: Receive x_i

2. Risk option: Receive x_n (the best consequence) with probability of π_i and x_1 (the worst consequence) with the complementary probability of $1-\pi_i$. The risk option will be denote as (x_n, π_i, x_1). Now the decision maker is consistent that he assigns $\pi_n=1$, $\pi_1=0$, so the π's are: $\pi_1<\pi_2<\ldots<\pi_n$. Now comparing $x_1<x_2<x_3<\ldots< x_n$ and $\pi_1<\pi_2<\ldots< \pi_n$ it can be seen that π's can be thought of as a numerical scaling of the x's.

The fundamental result of utility theory is that the expected value of the π*'s* can also be use to numerically scale probability distributions over the *x's*. Now by associating each x_i its scale π_i value then the expected π is as follows:

$$E(\pi') = \Sigma_i\, p'_i\, \pi_i \quad \textit{and} \quad E(\pi'') = \Sigma_i\, p''_i\, \pi_i$$

Now transform the π *'s* into *u's* by means of a positive linear transformation. Where $u_i = a + b\,\pi_i$, $b > 0$ *and* $i = 1, \ldots, n.$ *and* $u_1 < u_2 < \ldots < u_n$. Then the probabilistic choice or the expected value of u will be:

$$E(u) = \Sigma_i\, p'_i\, u_i = \Sigma_i\, p'_i\, (a + b\,\pi_i) = a + b\{E(\pi')\}$$

2.3 Metaheuristic approaches

In recent years, the meteheuristic approaches such as Tabu Search, Ant Colony, Particle Swarm, Simulated Annealing and Genetic Algorithms have been widely used with successful applications in different areas, such as: optimization in scheduling, facility layout, supply chain management, maintenance policy selection, spare parts inventory, assembly line planning, among others (Ali & Tunali, 2007). A description of each of them is provided below.

2.3.1 Tabu Search

According with the dictionary the Tabu word means "*A prohibition imposed by social custom as a protective measure.*" The basic concept of Tabu Search (TS) was developed by Glover in 1986, and has been used to solve a lot of NP-hard optimization problems such as, shop scheduling, the traveling salesman problem and the capacitated arc routing problem, among others. TS is a metaheuristic that guides or

helps another heuristic search procedure to explore the solution space beyond local optimum by use of a Tabu list. The main point of this approach is to avoid entertainment in cycles by forbidding or penalizing moves which take the solution in the next generation to points in the solution space previously visited (Glover & Laguna, 1997).

The importance of Tabu Search is reduced to its short term memory process, and many of the strategic considerations underlying this process reappear, amplified in degree but not greatly changed in link, in the longer term memory process. The short term memory and its aggressive search constitute a form of aggressive exploration that seeks to make the best highest evaluation, subject to requiring available choices to satisfy certain constraints. These restrictions operate in several forms, by direct exclusion of search alternatives and also by translation of modified evaluations and probability of selection (Glover, 1990). The restrictions are imposed by making reference to memory structures that are designed for this specific purpose. In other words, it has a flexible memory structure in conjunction with strategic restrictions and aspiration levels.

The Tabu search begins by showing local optima. To avoid repeating the steps used this method records recent move in one or more Tabu lists. The original target of the list was not to prevent a previous move from being repeated, but rather to guarantee that it was not reversed. The Tabu list will be recorded in the memory. The role of the memory can change as the algorithm proceeds. At the beginning, the main objective is to make an examination of the solution space, known as “diversification”, but as the locations are identified, the search is more focused to produce local optimal solutions in a process of

“intensification”. There are many cases in which the differences between the various implementations of the Tabu method have to do with the size, variability, and adaptability of the Tabu memory to a particular problem domain (Glover, 1990).

The use of memory

The memory of Tabu Search is based on the idea of problem solving in order to qualify as intelligent. Thus, it must incorporate adaptive memory and responsive exploration. The memory structures operate with four principal dimensions, consisting of *recency, frequency, quality* and *influence*. *Recency-based* and *frequency-based* complement one to each other, while the *quality* dimension refers to the ability to distinguish the value of solutions visited during the search. Thus, memory can be used to identify elements that are common to good solutions or paths that lead to good solutions. *Influence*, considers the impact of the choices made during the search, not only on quality but also on structure. The memory used is *explicit* and *attributive*. *Explicit,* because it records all the solutions during the search. It also records highly attractive but unexploited neighbors of elite solutions. This elite solutions are used to expand the local search, and in some cases to stay away from visiting solutions more than once. On the other hand it uses *attributive* memory for guiding purposes, it record information about solution attributes that change when in moving from one solution to another (Glover & Laguna, 1997; Glover, 1990).

This type of memory records information about solution attributes that change in moving from one solution to another (Glover, 1989). The efficiency of iterative solution methods depends mostly on the modeling. A well regulation of parameters will never

balance a bad choice of the neighborhood structure or of the objective function. On the opposite, an effective modeling should lead to robust techniques that are not too sensitive to different parameter settings.

Methodology

In order to improve the efficiency of this exploration process, it is needed to keep tracking not only of the local information but also of some other information related to the exploration process. This is a systematic use of memory characteristic of the Tabu Search. While most of the exploration methods keep in memory the value $f(i^*)$ of best solution i^* visited so far, Tabu search will keep the information on the schedule though the last solutions visited. This information will be used to guide the move from i to the next iteration j to be chosen in $N(i)$. It is easy to notice that the structure of the neighborhood $N(i)$ of a solution i will be in fact variable from iteration to iteration. In other terms in an optimization problem it could be seen as: S the set of feasible solutions and $f: S \rightarrow R$, find some solution i^* in S such that $f(i^*)$ is acceptable with respect to some criterion. Generally, the criterion of acceptability for a solution i^* would be $f(i^*) \leq f(i)$ (in case of minimization) or $f(i^*) \geq f(i)$ (for maximization) for every i in S. In such situations, Tabu Search is an exact algorithm that is going to provide the exploration process, and would guarantee that after a finite number of iterations the i^* would be reached (Glover & Laguna 1997).

The classical methodology proposed by Glover is as follows:

1. Choose an initial solution i in S. Set $i^*=i$ and $k=0$.

2. Set $k=k+1$ and generate a subset V^* of solution in $N(i,k)$ such that either one of the Tabu conditions is violated or at least one of the aspiration conditions.

3. Choose a best j in V^* (with respect to f or to the function f) and set $i=j$.

4. If $f(i) < f(i^*)$ then set $i^*=i$.

5. Update tabu list and aspiration conditions.

6. If a stopping condition is met, then stop. Else go to Step 2.

Where:

- $V^*=N(i)$ V^* may often be a substantial improvement.
- k is larger than the maximum number of iterations allowed
- $N(i,k)$ implies that some recently visited solutions are removed from $N(i)$; they are considered as Tabu solutions which should be avoided in the next iteration.
- T, tabu list
- m, Tabu move

Some immediate stopping conditions could be the following:

1. $N(i, K+1) = 0$. (no feasible solution in the neighborhood of solution i)

2. k is larger than the maximum number of iterations allowed.

3. The number of iterations since the last improvement of i^* is larger than a specified number.

4. Evidence can be given than an optimum solution has been obtained.

A general flowchart of the Tabu Search algorithm is shown below.

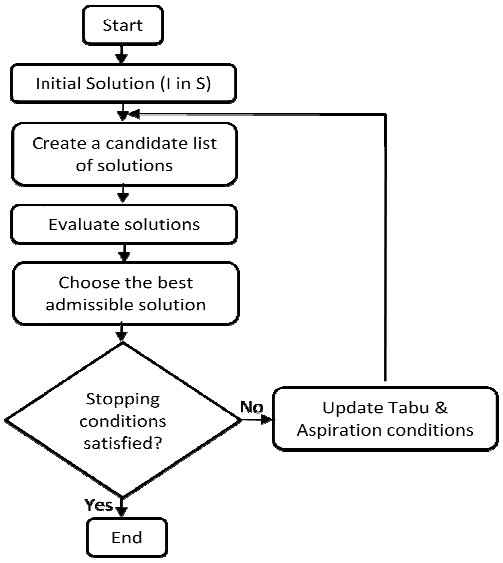


Figure 1. Tabu Search algorithm flowchart

2.3.2 Simulated Annealing

Simulated Annealing (SA) is a simple and general algorithm for finding global minima. SA is a random search technique which exploits an analogy between the way in which a metal cools and freezes into a minimum energy crystalline structure and the search for a minimum in a more general system. The analogy works because physical systems occupy only the states with the lowest energy as the temperature tends to absolute zero (Laarhoven & Aarts, 1987; Salamon *et al*., 2002).

Simulated annealing was developed by a wide and highly interdisciplinary community and it has been proved that by paying close attention to the rate of cooling and temperature, this approach can find the global optima, but requires an infinitive time. SA is an efficient optimization method that uses two kinds of techniques: first, it finds the global maximum by searching in new and unknown area in the space, and by knowing the points previously visited in order to find better points (Bertsimas & Tsitsiklis, 1993).

To determine whether the SA is the right tool to use, first it is need to consider the following aspects:

1. Determine whether local minima are a problem. It involves a huge search in a random sample of initial states. If the results are widely different, then it probably pays to do a careful search for global as opposed to just local minima.

2. Determine whether the problem warrants exploiting additional structure. This often offers information that already exists concerning the problem and algorithms for its solution and how easy the information can be implemented in the global optimization.

Methodology

The major advantage of simulated annealing over other methods is that it avoids to become trapped in local minima. This algorithm employs a random search that not only is going to accept changes that decrease the objective function, but also changes that increase it (Bertsimas & Tsitsiklis, 1993). The probability of acceptance is:

$$p = exp\ (\delta f/T)$$

Where :

- δf is the increase of f
- T is a control parameter which by analogy is known as Temperature.

In order to start there are some elements that must be provided:

- A representation of the possible solution.
- A generator of random changes in solution.

- A means of evaluating the problem function.
- An annealing schedule which contains an initial temperature and rules for lowering it as the search processes.

While solving an optimization problem using the SA algorithm, the way in which new solutions are generated may need some considerations. The solution generator should: introduce small random changes, and allow all the possible solutions to be reached. The flowchart of the SA is shown is Figure 2.

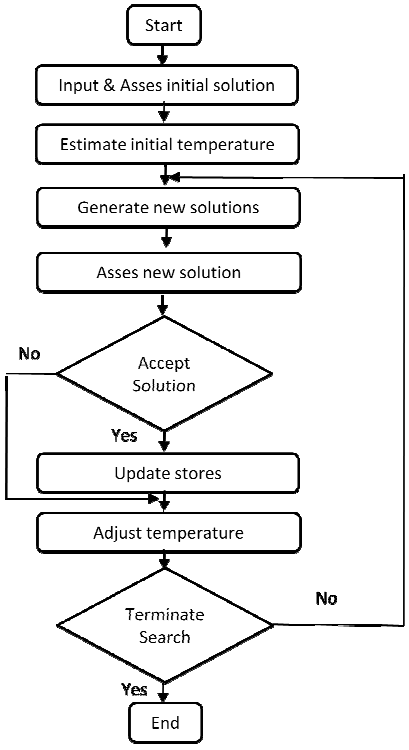


Figure 2. Simulated Annealing algorithm flowchart

2.3.3 Ant Colony Optimization (ACO)

Ants exhibit a complex social behavior that for a long time have attracted the attention of human beings. One of the most notable aspects of ACO is the fact of how ants construct what we call ant streets, which easily reacts to several disturbances and

obstacles during the track of the ant highways. There are certain ant species with surprising behavior patterns that have the ability to find what computer scientists call the shortest path. Multiple experiments have shown that this is possible by taking advantage of their communication based on pheromones. This behavior is what inspired researches to develop algorithms for the solution of hard optimization problems. Ant colonies can accomplish very complex tasks that in some cases exceed the individual capabilities of a single ant. The ant algorithm is derived directly from the study of ant behaviors, and uses this model as a source of inspiration for the design of algorithms and the solution of optimization problems (Dorigo & Stutzle, 2004).

The main idea inspired by the behavior of real ants, is the parallel search (certain ant species look for food in parallel threads, one taking care directly of the food, when the other, looks for a shorter and easier way to get hit target). This is compared over several constructive computational threads based on local problem data and on a dynamic memory structure containing information on the quality of previously obtained result. The collective behavior emerging from the interaction of the different search threads has proved effective in solving combinatorial optimization problems (Onwubolu & Babu, 2004).

Ant Colony Optimization algorithms are typically used to solve minimum cost problems. It usually has *N* nodes and *A* undirected arcs with two working modes for the ants: either forwards or backwards. Pheromones are only deposited in backward mode. The ants memory allows them to retrace the path it has followed while searching for the destination node. Before moving backward on their memorized path, they eliminate any

loops from it. While moving backwards, the ants leave pheromones on the arcs they traversed. The ants evaluate the cost of the paths they have traversed, so the shorter paths will receive a greater deposit of pheromones. An evaporation rule will be tied with the pheromones, which will reduce the chance for poor quality solutions.

At the beginning of the search process, usually started with a random solution, a constant amount of pheromone is assigned to all the arcs. When located at a node *i*, an ant *k* uses the pheromone trail to compute the probability of choosing *j* as the next node:

$$p_{ij}^{k} = \begin{cases} \dfrac{\tau_{ij}^{\propto}}{\sum l \in N_i^k} \tau_{il}^{\propto}, \textbf{if } j \in N_i^k \\ \mathbf{0}, \qquad \textbf{if } j \in / N_i^k \end{cases}$$

Where N_i^k is the neighborhood of ant *k* when in node *i.* When the arc (*i*,*j*) is traversed, the pheromone value changes by giving proportion:

$$\tau_{ij} \leftarrow \tau_{ij} + \Delta\tau^{k}$$

By using this rule, the probability increases that forthcoming ants will use this arc. After each ant *k* has moved to the next node, the pheromones evaporate by the following equation to all the arcs. The iteration is a complete cycle involving ants' movements, pheromone evaporation, and pheromone deposit.

ACO-based solvers systematically scan the set of possible solution elements before choosing a particular one. Due to this, the computational time required to get a solution from the several number of iterations from the algorithm can be large. The easier way to deal with this is to limit the number of elements chosen to a subsystem of candidate set.

This will help to find a competitive solution to the test problem in a relative short amount of time (Dorigo *et al.,* 2002).

This kind of algorithms has being successfully applied to benchmark problems such as the traveling salesman, the job sequencing problems, and also in the quadratic assignment problem, among others. In addition to more complex problems that have difficult constraints in areas such as transportation and telecommunications. Unfortunately, ant colony optimization techniques can suffer from long runtimes if attention is not paid to contracting appropriate subsets of elements from wish to choose. The essential purpose of ACO algorithms is the combination of the priority information about the structure of a promising solution with a posteriori information about the structure of previously obtained good solutions (Onwubolu & Babu, 2004). The flowchart of the Ant Colony is shown in figure 3.

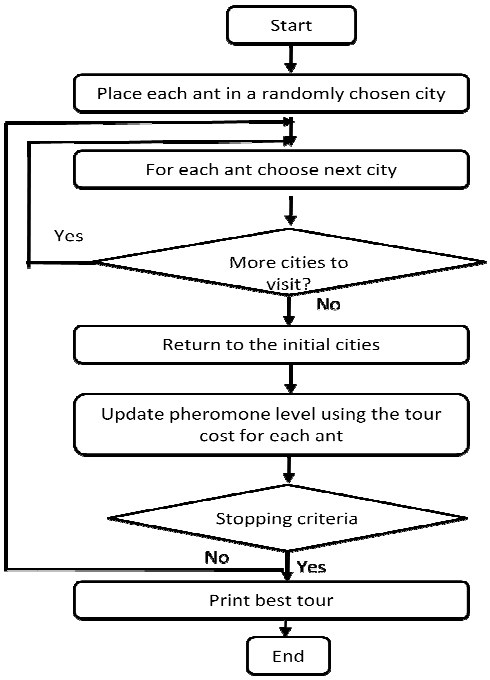


Figure 3. Ant Colony Optimization flowchart

2.3.4 Particle Swarm Optimization

The Particle Swarm Optimization algorithm (PSO) was proposed by James Kennedy and Russell C. Eberhart in 1995. It is mainly motivated by the social behavior of organisms such as bird flocking and fish schooling. PSO mimics the collective intelligent behavior of “unintelligent” creatures (Hu, 2006; Hu & Eberhart, 2002).

The swarm optimization method is a population based method just as Genetic Algorithms but instead of fighting one against the other, its concept is about mutual cooperation. PSO, has roots in artificial life and social psychology as well as engineering and computer science, and it differs from evolutionary computation methods in that the population members, called particles, are flown through the problem space, particles can be seen as an agent that flies all along the search space trying to find the best solution. This method has been successfully used in problem solving optimization with continuous search spaces.

Furthermore, PSO has been successfully applied in many research and application areas. PSO has been successfully implemented in many different types of optimization problems, such as in movie effects, neural networks, ingredient mix optimization, pressure levels optimization, swarm robotics and optimization of electric power distribution networks, among others. According to Kennedy, in terms of social and cognitive behavior, it is demonstrated that PSO gets better results in a faster, cheaper way compared with other methods such as Ant Colony Optimization, Genetic Algorithms and Neural Networks (Hu, 2006).

This method also provides evidence for theoretical perspectives on mind, consciousness, and intelligence.

Principles

- *Proximity principle:* The population should be able to carry out simple space and time computations
- *Quality principle*: The population should be able to respond to quality factors in the environment
- *Diverse response principle*: The population should not commit its activities along excessively narrow channels
- *Stability principle*: The population should not change its mode of behavior every time the environment changes
- *Adaptability principle*: The population must be able to change behavior mode when it is worth the computational price

Methodology

Particle Swarm Optimization algorithms optimize an objective function by conducting a population-based stochastic search. The population comprises potential solutions called particles, which are a metaphor of the birds flocking and fish schooling mentioned before.

The population is initialized by assigning random positions and velocities; potential solutions are then flown through hyperspace. Each particle (or agent) evaluates the function to maximize at each point its visits in spaces and updates its velocity and position based on the best experience of its own. Each agent needs to remember the

best value of the function found so far by its (*pBest*) and its coordinates. Secondly, the updating rule will steer the particle swarm to move towards the more promising region with higher objective value, and eventually all particles will accumulate around the optimum point. Each agent knows the global best position that one member of the flock had found and its value (*gBest*). At each time step, each particle stochastically accelerates toward its *pBest* and *gBest* (or *lBest*). The iterations are performed until a maximum number of iterations are reached or a minimum error criterion is met (Kennedy *et al*., 2001; Onwubolu & Babu, 2004).

Each particle keeps track of its coordinates in the problem space, which are associated with the best solution (fitness) it has achieved so far. This value is called *pBest.* Another "best" value that is tracked by the particle swarm optimizer is the best value obtained so far by any particle in the neighbors of the particle (solutions near to it in the search space); this location is called *lBest*. When a particle takes all the population as its topological neighbors, the best value is a global best and is called *gbest*. Once the algorithm finds the best values for *pBest* and *gBest*, the update process for the velocity and position of each solution is performed applying the velocity formula (Hu & Eberhart, 2002).

Velocity calculation:

$$v_{id_{(t)}} = \omega v_{id_{(t+1)}} + c_1 \times rand(*) \times (p_{id} - x_{id}) + c_2 \times rand(*) \times (p_{gd} - x_{id})$$

This can be easily explained as follows:

$$v[*] = v[*] + \{c_1 \times rand(*) \times (pbest[*] - present[*])\} + \{c_2 \times rand(*) \times (gbest[*] - present[*])\}$$

Personal influence *Social influence or imitation*

Where:

- *v*[*] is the particle velocity
- *present* [*] is the current particle (solution).
- *pBest*[*] is Particle best position
- *gBest*[*] is global best position
- *rand* (*) is a random number between (0,1)
- c_1, c_2 are learning factors; usually $c_1 = c_2 = 2$.

From the equation above it can be seen that the terms that belong to the first {} space correspond to the personal influence of the particle's previous knowledge, while the second group of {} affects the particle velocity according to the social influence and the ability to imitate it.

One of the most important things in PSO is that in this algorithm a particle never dies. Every time the velocity of each particle is updated, also the position of it has to be recomputed. These new positions are gotten using the following equation:

$$x_{id_{(t)}} = x_{id_{(t+1)}} + v_{id_{(t)}}$$

In other terms:

$$present\,[*] = present\,[*] + v[*]$$

The same notation from the velocity equation applies for the position equation.

At each iteration the values of *pBest* and *gBest* are also revised to detect if a particle found a better solution than the one from the previous iterations, and as it was mentioned, the algorithm iterates as many times as it is established by the analysts or until a difference from an objective value is minimum and accepted.

The flowchart of the PSO algorithm is shown in figure 4.

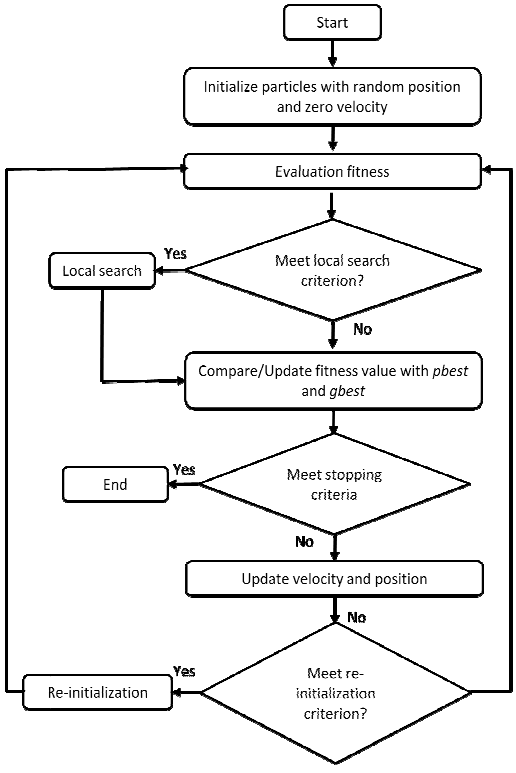


Figure 4. Flowchart for the PSO algorithm

2.3.5 Genetic Algorithms.

A Genetic Algorithm (GA) is another metaheuristic approach biologically inspired, useful to solve NP-hard optimization problems (Ali & Tunali, 2007). It starts with the generation of a random population which represents potential solutions for the specific problem. When the population is randomly generated, the algorithm evolves by making use of the main operators:

(1). *Selection.* Equates to survival of the fitness. The selection operator gives preference to the better individuals, allowing them to pass their genes to the next generation. The goodness of each individual depends on its fitness, this may be determined by the objective function or by a subject judgment.

(2). *Crossover.* Represents mating between individuals. It's the factor which distinguishes GA from other algorithms. Two individuals are chosen from the population using the selection operation. A crossover site along the bit strings is randomly chosen. The two values of the strings are exchanged up to this point (as shown in figure 5). The two new offspring are put into the next generation of the population. By recombining these individuals, this process is likely to create better individuals.

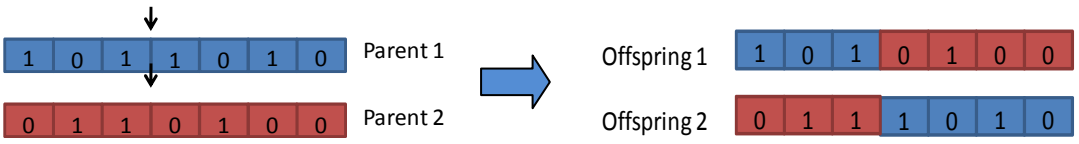


Figure 5. A one-point crossover example

(3). *Mutation.* It introduces random modifications. Its purpose is to maintain the diversity within the population and inhibit premature convergence. It introduces a

random walk through the search space. Mutation and selection (without the crossover) create a parallel, noise tolerant algorithm.

Facts of the genetic operators:

- By using just the selection operator will tend to fill the population with copies of the best individuals from the population.
- By using the selection and crossover operators will tend to cause the algorithm to converge on a good but sub-optimal solution.
- If using mutation by itself introduces a random walk thought the search space.
- Using selection and mutation creates parallel noise tolerant, hill climbing algorithm.

General Methodology of a single objective GA

1. Randomly initialize population
2. Determine the fitness of population
3. Repeat

3.1 Select parents from population

3.2 Perform crossover on parents creating population

3.3 Perform mutation of population

3.4 Determine fitness of population

4. Stop until stopping criterion is met.

Figure 6 presents the flowchart of a single-objective GA.

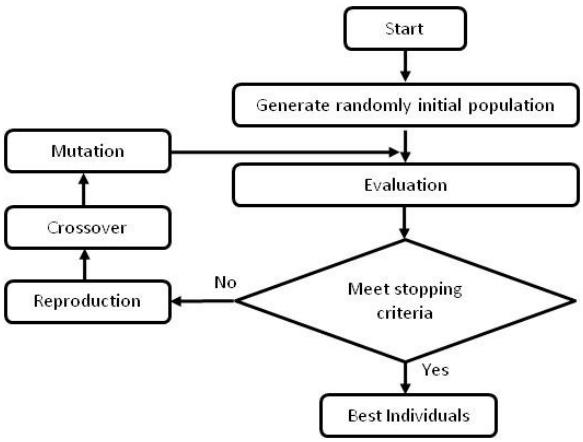


Figure 6. General Genetic Algorithm flowchart

On next chapter a completed and more detailed explanation of this latest algorithm is described.

Chapter 3 provides a detailed explanation on single and multiple objective GAs.

CHAPTER 3. SINGLE AND MULTIPLE OBJECTIVE GENETIC ALGORITHMS

3.1 Introduction

The Genetic Algorithms (GAs) were developed in the 1960's by John H. Holland at the University of Michigan. Similar to others biologically inspired techniques such as ant colony, particle swarm, etc. Genetic algorithms take out the idea from the natural evolution system. They are problem independent algorithms that can process the information generated in previous iterations or stages (Cheng *et al.,* 1997). This algorithm is inspired by evolution and it is encoding a potential solution to a specific problem on a simple chromosome or structure by applying different combinations to such structure. As in all optimizations problems the purpose is to minimize/maximize the objective function subject to some constrains (Whitley, 1994).

The working mechanism of a genetic algorithm can be summarized as follows. First, a random population of solutions is generated. The goodness of the solution is defined with respect to the current population (Konak *et al.,* 2006). Next, the best solutions are recombined with each other to form some new solutions. Finally, the new solutions are used to replace the poorest solutions, and then the process is repeated for a specific number of generations (Paiton & Campbell, 1995; Whitley, 1994).

In Genetic Algorithms, the solution candidates are called individuals; each individual represents a possible solution, a decision vector to the problem, and the set of solution candidates is called the population, the term chromosome refers to the solution to a problem (Chiu *et al.,* 2006). The genes are either single bits or short blocks

of adjacent bits that encoded a particular element of the candidate solution. The crossover is typically considered the exchanging genetic material between two single chromosome parents. Mutation consists of flipping one or more bits at randomly chosen locations.

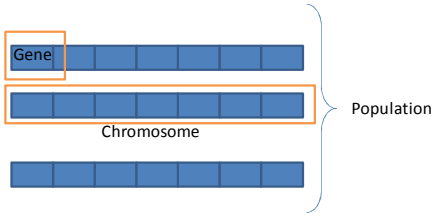


Figure 7. Genetic Algorithms basic concepts

In the real world, decision making generally involves simultaneous optimization of multiple objective functions instead of a single one. In single objective optimization, GAs attempt to obtain a “global goal”. However, in the case of multiple objective optimization, most of the time a best solution does not exist or is harder to attain, instead, it ends up with a set of solutions which are the non dominated solutions to the problem that are superior to the rest of them. When considering this set of solutions, some solutions can be inferior in some aspects, but superior in some others.

When dealing with multiple objective optimization problems, most of the traditional methods assign weights to the objective functions to create a single objective. For those cases, there is a big chance that the obtained solution can be far away from an optimal solution, therefore the need to consider multiple solutions instead of a single one. Genetic Algorithms provide a large set of possible solutions that represent a good alternative in multiple objective optimization problems, since they work with a population set of points (multiple solutions). These solutions are shown in Pareto front as Pareto optimal solutions or nondominated solutions (Martinez *et al.,* 2009). The choice of one

solution over any other is a hard decision, since none of the nondominated solutions are absolutely better than any other.

3.2 Pseudo code of a Genetic Algorithm

The pseudo code of a general single objective GA can be summarized as follows:

1. Generate a random population of *n* chromosomes which represent possible solutions.

2. Establish a method to evaluate the fitness *f*(*x*) of each chromosome *x* in the population.

3. Create a new population by repeating the following steps until the new population is complete.

    a. *Selection*: Select from the population according to their fitness.

    b. *Crossover:* Create new offspring formed by a crossover with the parents.

    c. *Mutation:* With a low probability of mutation, randomly selected offspring.

4. Use the newly generated population for a further run of algorithm

3.3 Search space

A population of individuals is maintained within search space for the Genetic algorithms. Each individual is represents a possible solution for a given problem. The individuals are coded as a finite length vector of components variables. The type of encoding of the solutions usually depends on the type of the problem to be solved. Each encoded solution is evaluated and the one with the highest objective function value represents the best solution. The GAs aim to use selective breeding of the solution to produce better offspring by combining information from the chromosomes.

The algorithm maintains a population of *n* chromosomes with their associated fitness values. Parents are selected in order to mate on the basis of their fitness value. As a result, highly fit solutions are given more opportunities to reproduce so that offspring inherit the characteristics from the parents. As parents reproduce, weak individuals in the population die and are replaced by the stronger solutions.

Those new generations of solutions are produced containing on average, more good genes that the previous generation. Each successive generation will have better partial solutions than the previous ones. The number of generations should be specified in the algorithm (Azaron *et al.,* 2009). Eventually, as the search evolves, the population includes fitter and fitter solutions and it converges meaning that is not producing different offspring than in previous generations, the algorithm itself is said to converge to a set of solutions to the problem at hand (Konak *et al.* 2006).

3.4 Encoding technique

The chromosomes contain information about the solution, the type of encoding depends on the problem, and it determines the complexity and requirements of the genetic operators. There are different kinds of encoding (Mitchell, 1996):

- Binary encoding

One way of encoding is using the binary string using bits of 0 or 1. Each bit in the string represents some characteristic of the solution or it could represent whether or not some particular characteristic is present.

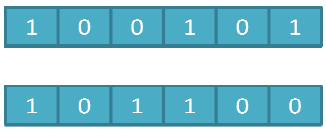


Figure 8. Binary encoding

- Permutation encoding

Permutation encoding can be used in ordering problems. Every chromosome is a string of numbers, which represents number in a sequence.

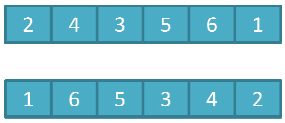


Figure 9. Permutation encoding.

- Value Encoding

Value encoding can be used in problems with complicated values, such as real numbers, where the use of binary encoding would not suffice.

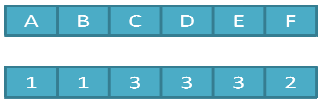


Figure 10.Value encoding

- Tree encoding

This kind of encoding is used to actually have programs or expressions evolve. In tree encoding every chromosome is a tree of some objects, such as functions or commands in the programming language.

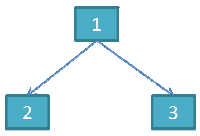


Figure 11.Tree encoding

3.5 Selection

This is the process in which individual chromosomes are copied according to their fitness. It is not just about measuring the profit or utility that is going to be maximized. Copying the chromosomes according to their fitness means that the chromosomes with a higher value have a higher probability of contributing one or more offspring in the next generation (Cheng *et al.,* 1997). There are many methods for selecting the best chromosomes such as: roulette wheel selection, Boltzman selection, tournament selection, rank selection, steady state selection and others (Mitchell, 1996).

- Roulette Wheel

Simple allocation of the offspring strings using a roulette wheel according to their fitness. It is the basic part of the selection process to stochastically select from one generation to create the next generation. The parents are selected according to their fitness. The better the fitness of the chromosome, the greater the chance it will be selected.

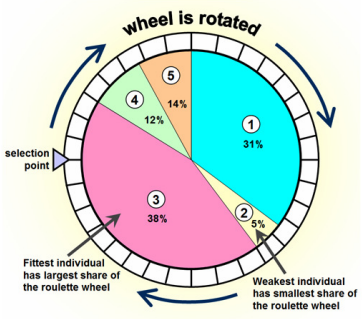


Figure 12.Roulette Wheel selection

- Steady State

The members of the population changed one at a time. In order to perform the selection a member will be chosen according to its fitness. It will be copied and the copy

will be mutated. Then a second member of the population is selected which is replaced by the mutated string. Then two members of the population are chosen and a single child is created which replaces a member of the population.

- Tournament

The tournament selection works by randomly selecting *n* individuals and the fittest value is evaluated and compared it. The one with the highest fitness value will be assigned to be a parent to perform the crossover.

- Elitism

The best chromosomes are copied to the population in the next generation. Elitism is a method that can very rapidly increase the performance of GAs, this is because it prevents losing the best solution during the optimization process due to random effects. A variation of this method could be the elimination of equal numbers of the worst solutions.

- Rank Selection

This is a variation of the roulette method of selection because it can present problems when the fitness has considerable differences. If the best chromosome fitness is 95% of the entire roulette wheel then the other chromosomes will have a small chance of being selected.

The rank selection method first ranks the population and then every chromosome will receive fitness from this ranking. The worst will have fitness 1, second worst 2 etc. and the best will have fitness *N* (number of chromosomes in population).

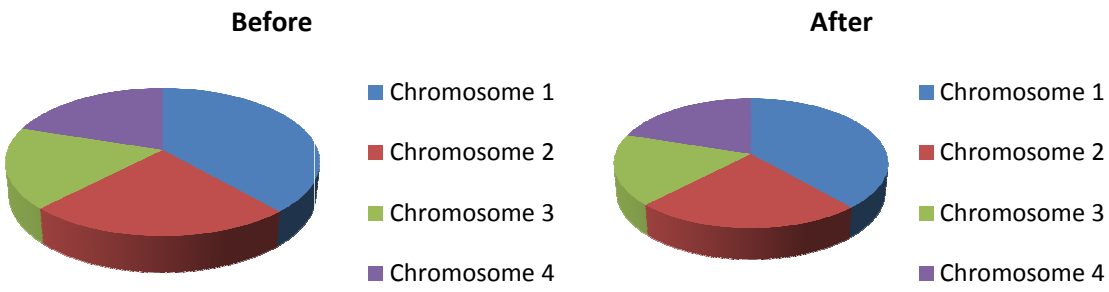


Figure 13.Rank Selection

3.6 Crossover

Once the selection of the individuals is done, it is time to create new offspring by performing the crossover (Reeves & Rowe, 2003). The effectiveness of this operator gives the rate of convergence of the algorithm. While there are many kinds of crossovers, the most common is the single point crossover. The variety of crossovers is shown below:

- Single point crossover

It consists in randomly selecting a point. The offspring will take one part of one of the parents and the rest from the other parent.

Figure 14.Single point crossover.

- Two point crossover

It works as the previous one, but in this case selecting two points randomly.

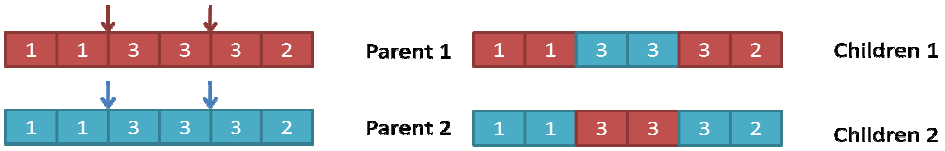


Figure 15. Two points crossover

- Multiple point crossover

As the previous ones, this type of crossover randomly selects different points to be swaped.

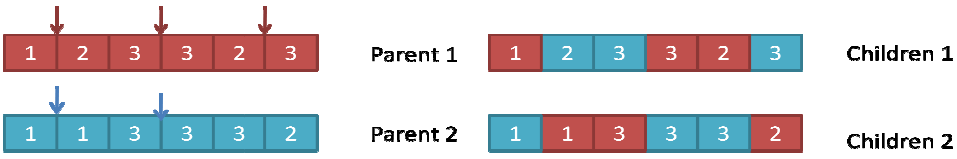


Figure 16. Multiple point crossover

- Uniform crossover

A certain number of genes are randomly selected to be exchanged.

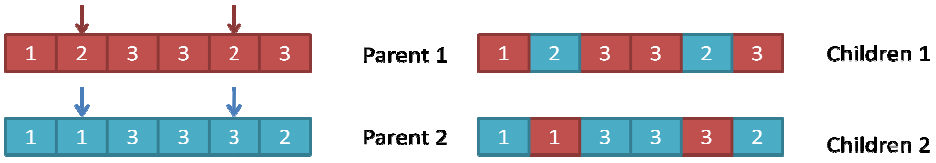


Figure 17. Uniform crossover

- Subsystem rotation crossover

Multi parent recombination is allowed and it creates a large number of children which could be translated into a large number of solutions that will be evaluated (Taboada & Coit, 2009).

To calculate the number of children, this equation is used:

$$^{\#}C = {}^{\#}S\,[{}^{\#}P\,({}^{\#}P\text{-}1)]$$

Where:

$^{\#}C$=Number of children

$^{\#}S$=Number of subsystems

$^{\#}P$=Number of parents

Here is an example of four chromosomes of three subsystems that were selected as parents. This is how it works for Subsystem 1.

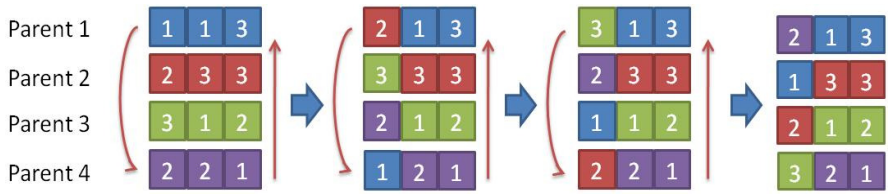


Figure 18. Subsystem Rotation Crossover

The same happened with the other subsystems creating a grand total of 36 children.

3.7 Mutation

After having a new population of individuals, some will be directly copied and others will be produced by the crossover. In order to avoid that the individuals are not exactly the same, the possibility of mutate them is one choice by flipping a bit of a chromosome. Mutation is usually occurs at a very low probability. Mutation represents a a chance to prevent premature convergence from occurring (Man *et al.,* 2001). A new chromosome will be created by random modification of some of the genes.

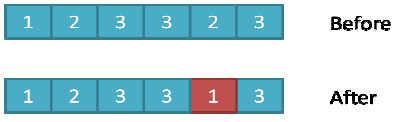


Figure 19. Mutation

The loop consisting on the steps of evaluation, selection, recombination and mutation is performed several times, each loop is called generation. The stopping criterion for this algorithm will be satisfied until the maximum number of generations is reached, other type of stopping criteria involve stopping once a maximum computing time has been reached or when the fitness function value remains unchanged during the last *n* generations. At the end the best individual in the final population represents the outcome of the algorithm (Gutierrez & Briones, 2009).

3.8 Applications

Genetic algorithms work in a very effective way of quickly finding a good solution for a complex problem. Genetic algorithms are used for optimization purposes by selecting the best alternative of a set of given options. Like in any optimization problem it works with an objective function or an objective that depends on a series of variables. GAs are excellent for any task that requires optimization, they are highly effective in:

- *Management:* Distribution, scheduling, task assignment, container packing, time tables, etc.
- *Financial:* Portfolio balancing, investment analysis, payment scheduling, budgeting, etc.
- *Engineering:* Structural, mechanical, network and electrical design, process control, etc.
- *Optimization:* Data fitting, clustering, trends and path finding.

Genetic algorithms are an excellent way to solve problems by mimicking the same processes as nature. They use the same combination of selection, recombination and mutation to evolve form a solution to a problem.

The benefits of using this method fall into an enormous list that shows why they are preferred over other methods.

- Easy concept.
- Supports multiobjective optimization.
- Good for noisy environments.
- The answer gets better each time.
- Flexible building blocks

3.9 Pareto Optimality for Multiple Objective Optimization

Most of the engineering problems turn into a multiple objective problem. The best way to find a good solution will be the simultaneous optimization of all of them even when they may be opposed to one another, that is when the improvement of one objective involves the deterioration of the others. Multiple objective optimization techniques offer an advantage over the single objective optimization techniques because they provide a set of optimal solutions instead of just one, all with different tradeoffs, where the final decision will be in charge of the decision maker.

The concept of the Pareto optimum or Pareto front was developed by Vilfredo Pareto in the XIX century, and it is considered the origin of the multiobjective optimization research (Gutierrez & Briones, 2009). When dealing with multiple

objectives, the solutions can be easily shown in the Pareto Front, where each point represents a good solution for the different objectives in conflict. In the Pareto front, there is not a point better than the others, since all of them are nondominated points. The engineer in charge will pick a point of the Pareto front that will be an acceptable tradeoff between the different objectives (Gutierrez & Briones, 2009).

The plot of the objective functions whose nondominated vectors are in the Pareto optimal set is called the Pareto front. In contrast to fully ordered scalar search spaces, multidimensional search spaces are only partially ordered, the different solutions are related to each other in the following possible ways (Fonseca & Fleming, 1993):

1. Inferiority or Dominated

A vector $u=(u_1, u_2, \ldots, u_n)$ is said to be inferior to $v=(v_1, v_2, \ldots, v_n)$ iff v is partially less than u ($v\ p<u$)

2. Superiority or Dominance

A vector $u=(u_1, u_2, .., u_n)$ is said to be superior to $v=(v_1, v_2, .., v_n)$ iff v is inferior to u.

3. Non-inferiority or Nondominated

A vector $u=(u_1, u_2, \ldots, u_n)$ and $v=(v_1, v_2, \ldots, v_n)$ are said to be non inferior to one and other if v is neither inferior nor superior to u.

This kind of point has the characteristic that when it is compared to any other feasible point in all the objective function space, at least one of its objectives functions

values is greater to the corresponding objective function value of this other feasible point.

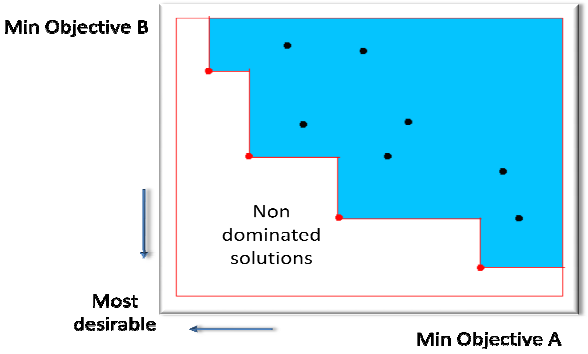


Figure 20. Pareto optimal frontier

By maintaining a population of solutions, Genetic Algorithms can search for many non-dominated solutions. This characteristic makes GAs very attractive for solving MO problems.

CHAPTER 4. PERFORMANCE BASED LOGISTICS

4.1 Introduction

Nowadays we are living in an era of competing global priorities, reduced acquisition budgets and thinly stretched support resources that are affecting not only commercial business but also the government agencies (Beggs *et al.,* 2005). In response to this, there is an area of focus that has been a shift from traditional transaction based support to Performance Based Logistics support (PBL). PBL is one of the best ways to optimize the cost of the procurement of goods and services. Table 1 shows the transition from business model element to PBL attributes.

Table 1. Transition from business model element to PBL attributes

Business model element	**Traditional attributes**	**PBL attributes**
Payment for delivering	Pay upon transaction	Fixed periodic payment
Warranty	Limited	None
Customer relationship required	Low level	High level
Contractor internal operational performance visibility	Opaque	Transparent
Contract length	Short; less than 2 years	High; usually more that 5 years

In the past, particularly the military agencies used to handle a total control for their logistic system by using an enormous support of the infrastructure based on maintenance concepts designed around full military repair responsibility (Smith, 2004). This traditional logistic support creates inefficiencies with the metrics that were focused just on internal logistics processes, and rarely have direct relationship with the warfighter requirements. The need of an improved logistic system, agile, flexible and able to respond to unpredictable demands leads the Department of Defense (DoD) to

create a new product support plan. Now they are using PBL contracts based on their needs, using incentives to improve readiness at an optimal cost and managing the risk to the commercial contractor.

4.2 Performance Based Logistics

In the past the DoD has dictated to the contractors what to produce, when to produce it and the activities that they should carried out. The more the contractor produced, the more money they made (Vitasek & Geary, 2008). However, in 2001 the DoD adopted a new approach in order to reduce operations and support for their weapon systems. PBL promises a break with the traditional approach which operates and maintain the military. PBL focuses on performance outcomes not in individual parts or repair actions. Now the government simply tells the contractor the desired performance outcome they want in terms of measurable metrics and let them use their best practices to efficiently and cost effectively meet the goals. With this new agreement the DoD does not have to worry about the payment for unit transactions, services as warehousing, transportation, spare parts, repairs or hours of technical support (Vitasek & Geary, 2008; Goure, 2009).

According to the DoD, “PBL is the purchase of support as an integrated, affordable, performance package designed to optimized system readiness and meet performance goals for a weapon system through long-term support arrangements with clear lines of authority and responsibility.” (Berkowitz *et al.,* 2005; Vitasek & Geary 2008; Gansler & Lucyshyn, 2006).

Performance Based Logistics is the new DoD preferred method of support which has gained popularity lately. It basically moves the focus from management of parts and supplies to management of the suppliers responsible for delivering the required performance, such as availability and response time.

The real meaning of PBL is the purchase of weapons system sustainment as an integrated, affordable package based on output measures such as the weapon system availability, rather than input measures such as parts and technical services. While delivering parts and supplies in the right time and quantities, buying performance outcomes instead of parts, goods, man hours or services translate into reducing cost, decreasing cycle times, improving performance and predicting demand. But PBL not only improves the weapon system availability, it also reduces the cost of sustainment, maintenance and support activities. A greater availability could be transformed into more equipment or platforms on hand for operators in the field (Giannotti *et al*., 2006). Some other objectives of PBL are the considerable reduction of the work in process, repair times and backorders.

PBL offers some other advantages among the traditional approaches; it reduces the demand for government personnel, provides lower cost alternative of maintenance by using the private sector methods and other people, reduces program cost through modification or use of existing information technology and tools, reduces down time, increases operational availability and creates support structure that is flexible and open to warfighter needs (Goure, 2009).

Some of the most important characteristics from this latest method are:

- The contractor shares risks and rewards to ensure availability.
- Motivate to continuously improve the reliability.
  - Less parts and maintenance.
  - Correct problems before they result in inventory.
  - Deals with obsolescence.

Operations and maintenance cost are increasing due to the obsolescence of the military's hardware, and at the same time, new complex and reliable systems are arriving and the cost of their maintenance is extremely high. The maintenance process must ensure the high readiness levels and availability which is essential for the platforms and weapons system (Price, 1991; Goure, 2009).

Availability is considered one of the most important criterions for repairable systems that account for both the reliability and maintainability of a specific component or system (Nowicki *et al*., 2007). And if it is considered the reliability and the maintainability, then an additional metric must be considered for the probability that such component or system is operational in a given time. The availability, is defined as the probability that a system is not failed or going to a repair action when it is actually needed. This metric is always associated with a period of time (Hou & Okogbaa, 2005).

This way, Performance based logistics empower the provider to decide how best to meet the objectives focusing on operational readiness rather than just delivering the product. The provider has the freedom to create a network of capabilities and initiatives

to achieve the performance, cost and customer satisfaction targets. The different objectives that PBL manages are:

- Increase customer satisfaction.
  - Increase system availability.
  - Increase system reliability.
  - Reduce the logistic footprint in order to increase the speed of deployment.
  - Decrease life cycle cost to manage a system.
- Increase the use of contractor's management.
- Reduce the transactional intensity associated with a legacy offering.
- Increase the duration of contractual agreements in order to provide the contractor opportunities to make long term investments that will:
  - Lower and stabilize their total expenditures.
  - Lower and less volatile prices.
  - Be more efficient and effective in system lifecycle management.

Sometimes is hard for PBL to be implemented mainly because of the presence of some of the following aspects:

- Lack of defined roles and responsibilities of both parts, customer and provider.
- Misalignment of support provider competencies with customer needs.
- Lack of relevant metrics. (How reliable is the equipment?, How often does it break down?, At what levels will repairs be required?)
- Inadequate data collection and interpretation process for logistic management.

4.3 Performance Based Agreement

All PBL implementations are unique. They are designed to fit into a specific system, and specific needs of the warfighter. Support on this, the mechanism by which Performance Based Logistics is implemented for a specific platform, system or item is

based on a Performance Based Agreement (PBA). This is a contract between the parties for end to end customer support which is mainly focused on clear expectations, goals, resources, rules, responsibilities, benefits, metrics, methods, etc. for all the elements of support covered by the agreement (Gansler & Lucyshyn, 2006; Goure, 2009). It requires a continuous flow of high quality information of every element in the supply chain and parts or systems subject to the contract; the performance levels and the support metrics will be documented on it.

The metrics on it should reveal the needs of the warfighter, they should be clear, and be expressed in terms of performance criteria that related to the desired outcomes i.e. cost, reliability, maintainability, and some others. This first portion of work is engineering based and it's needed to develop measurable equipment, system, or platform performance metrics. This information allows the PBA contractor to anticipate the demand, implement changes in the design, fabrication or transportation.

Usually, these kinds of contracts last ten or more years and it is reviewed every five years, in which the supplier takes the control of a significant portion, or all, of the supply chain and guarantees the agreed level of performance. The supplier could be able to appreciate the costs involved in supporting a platform or system as well, identify the opportunities for cost reduction and implement the necessary changes. The provider needs time to make the necessary investment and see the return of it. PBL contracts also reduce the total cost of ownership because they can be 15 times longer than a traditional service and repair agreement (Keating & Huff, 2005; Goure, 2009).

To date, most of the PBA have been successful; they have increased the availability of the equipment and systems of the military in combat. PBA also appear to be saving money from the government, with just 23 contracts the average savings goes up to $21 million dollars (Goure, 2009).

4.4 Performance Based Logistics Supply Chain

Now a description of each level of the PBL supply chain is presented:

- Warfighter/Customer/Organization. Refers to the one who required support from an organic contractor logistic infrastructure to meet their objectives. It establishes the detailed requirements for a PBA, specified the required performance, provides feedback to the program manager and also funds support. It needs reliable support to meet its objectives, doesn't matter where it comes from as long as it has it, the last thing that they would like to do is deal with all of the contractors, because they are already busy.
- Program manager/Product manager (PM). It refers to the leader for developing, fielding and supporting. It is the one that negotiates the terms of the PBA with the customer. It is responsible of supporting the total life cycle, and for the execution of the PBA. It is directly accountable to the customer support in order to improve the system availability, optimize the support, improve the system readiness, reduce costs, reduce logistic footprint. Also, it is directly accountable to the rest of the supply chain in order to ensure its optimization, stable and reliable throughout the lifecycle of the product.
- Product support integrator (PSI). It is the team responsible for integrating those aspects for logistics necessary to achieve the goals of the PBA. It manages the system supply chain in order to reach availability, cost objectives and logistic footprint reduction.

- Product support provider (PSP). It delivers support and is compensated based on achievement of negotiated performance metrics derived from the PBA.

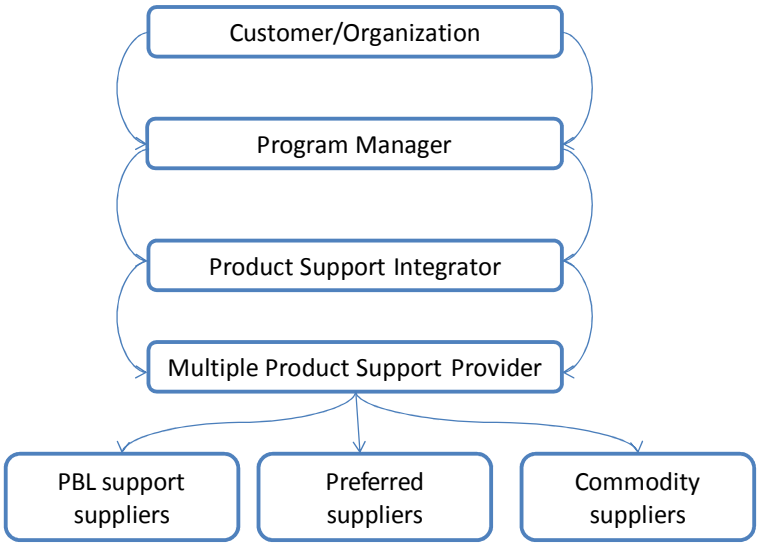


Figure 21. PBL Supply Chain

PBL has transformed the supply chain, giving as a result a win-win situation for customers and suppliers, PBL is the key of success between the relationship of the provider and the customer. PBL changes the risk that exists between the supplier of services and the government (Keating & Huff, 2005). PBL has been used for years for commercial business in order to support their sales of heavy machinery, generators and engines, such as, Caterpillar, Honeywell, Rolls-Royce, and Allison (Goure, 2009).

Recently, critics of PBL have argued that this approach is more expensive than the traditional one, and that increases are obvious on contractor profits based on an unfair arrangement. But these critics fail to consider the real needs of the warfighter. Those critics do not accept that the contractor profit depends on delivering a mix of improved availability and lower cost. It is clear the extraordinary impact that PBL has on the cost and performance of sustainment and support activities. There is enough evidence that

shows the use of PBL in other countries like in the United Kingdom's Ministry of Defense which supports critical systems (Goure, 2009).

4.5 Performance Based Logistics Objectives

The most important objectives for the DoD is to gain quality in their products/systems that satisfy their user needs with measurable improvements in a considerable time and at a fair cost. At this point, it is easy to see the importance and the magnitude of PBL and how the different objectives need to be optimized simultaneously, particularly those concerning to the system design, such as the reliability and the maintainability, especial elements for the mission capability. These two values are the key in a weapon system; their importance has been recognized as a significant factor for the Army combat units. The influence of the reliability and maintainability in a decision is in the top level over other factors.

There are an extensive variety of parameters to express the weapon system requirements; the engineers in charge have used them to describe and demonstrate or simply predict characteristics of the system. The Reliability has to do with the quality of the measurements. Reliability is defined as the probability that a component or system will perform a required function for a given period of time when used under stated conditions. One of the main purposes of the reliability analysis is the identification of the weaknesses in a system and the quantification of the impact of the component failures. Reliability can take values between one and zero. (Ebeling, 1997; Dumma & Krieg, 2005). In an exponential distribution system the reliability can be obtained by:

$$R = e^{-\lambda t}$$

- If Reliability = 1 means a perfectly reliable system.
- If Reliability = 0 means a perfectly unreliable system

The reliability is an important product attribute, including:

- *Reputation*. This is related with the company's reputation linked to the reliability of their products. The more reliable a product is the more favorable reputation the company has.
- *Customer Satisfaction*. Always an unreliable product will negatively affect customer satisfaction severely, while a reliable product may not affect it in a positive manner. A high reliability is a requirement for customer satisfaction.
- *Warranty Costs*. Warranty is an obligation to products that requires manufacturers to provide to their customers when the products fail to perform their designed functions under normal usage within the warranty coverage. The replacement and repair costs will negatively affect profits. Introducing reliability analysis is an important step in taking corrective action, ultimately leading to a product that is more reliable.
- *Repeat Business.* An improved reliability shows to customers that a manufacturer is serious about their product, and loyal to customer satisfaction. This attitude has a positive impact on future business.
- *Cost Analysis*. It is the combination between the reliability data and other cost information to illustrate the cost-effectiveness of their products.
- *Competitive Advantage*. Many companies use to publish their predicted reliability in order to gain an advantage over other companies.

Sometimes the product has a reliable design but when the product is manufactured and used in the field, its reliability may be not accepted. Some of the reasons for this low reliability may be poor manufacturing. So, while this product may have a reliable design, its quality is not acceptable because of the manufacturing process.

In technical terms, reliability expresses the life units between the operational mission failure and the probability for accomplishing a given mission. How frequently during a mission does a soldier expects his weapon to fail? By improving the system reliability of the weapon, the risk of a mission failure decrease (Price, 1991).

The Maintainability is defined as the probability that a failed component or system will be restored or repaired to a specified condition within a period of time when maintenance is performed in accordance with prescribed procedures (Ebeling, 1997). When is performed by personnel having specific skills levels, using the right procedures and resources, it measures the simplicity and velocity in which a system can be restored to the operational status after a failure has occurred. The prediction of the Maintainability allows defining the repair tasks and easily reusing this information for all the design. For instance, assume that a component has 95% of maintainability in one hour, this means that there is a 95% probability that such component will be repaired within an hour. In maintainability, the random variable is the Time to Repair, in the same manner as Time to Failure is the random variable in reliability. (Ebeling, 1997; Dumma & Krieg, 2005). Maintainability can be given using the following formula:

$$M(t) = 1 - e^{-\mu t}$$

The Mean Time To Repair (MTTR) is the average time required to perform corrective maintenance on all of the removable items in a product or system. This kind of prediction analyzes how long repairs and maintenance tasks will take in the event of a system failure, and can be obtained by:

$$MTTR = \frac{1}{\mu}$$

The reliability and maintainability are considering complementary measures in the calculation of the inherent availability.

$$A_i = \frac{MTTF}{(MTTF + MTTR)}$$

- If the MTTF is a large number compared with the MTTR, then a high availability is obtained.
- If the MTTR is a small number, then the availability will be high.
- As reliability decreases, a better maintainability is needed in order to achieve the same availability.

The operational world considers other measure known as operational availability, and differs from the previous one since this one considers the Mean Time Between Maintenance that includes all corrective and preventive actions and the Mean Time To Failure which only considers failures.

$$A_O = \frac{MTBM}{(MTBM + MDT)}$$

The Mean Down Time (MDT) refers to the system being down for the corrective maintenance, including delays, in contrast with the MTTR which only considers the repair time and the number of failures from a particular period of time. It indicates that the machine is down for a long period of time. It can be due to spares shortages, long waiting time (queue) delayed diagnosis, etc. It is expressed in time units.

$$MTD = \frac{Total\ down\ time}{Number\ of\ breakdowns}$$

The main difference between the MTBF and the MTBM is that the MTBM includes all corrective and preventive actions and the MTBF only take into account the failures. It is also undoubtedly influenced by logistics considerations such as spare parts, personnel, strategic resources, test equipment and tools, which are customer dependent. The reliability and maintainability are characteristics provided for the design of the product.

Achieving specific levels of reliability and maintainability of a system is very important because it has an effect on the total cost of ownership. The total cost of ownership attempts to capture the real cost for the design, development and support for the DoD weapon system. It defines the sum of all financial resources necessary to organize, equip, and sustain the military forces. Operating and support costs are the most common fees visible to the management than any other factor. Money is and will be the common denominator for a person to understand the technical and nontechnical concepts. In the Pentagon and in the Congress the importance of the weapon system cost is unquestionable. In fact the cost associated with the weapon system is a

significant factor, but not the driver. Reductions in money are significant but there is no need to let aside the reliability and the maintainability. Decisions must take into account these three concepts in order to improve the weapon system, to enhance the combat effectiveness (Price, 1991).

In the developing stage weapon systems requirements are mainly focus on technical performance, with a small attention in operation and support cost and readiness, using special technologies to meet performance goals that assure high reliability and maintenance.

On the present thesis, the problem to solve and optimize pertain to the design stage and it involves multiple objectives, thus the use of an Evolutionary algorithm seems particularly useful to solve this kind of problems because they deal simultaneously with a set of possible solutions. An evolutionary algorithm allows to find several members of the Pareto optimal set within a single run of the algorithm, instead of having to perform a series of separate runs as in the case of the traditional mathematical programming techniques. Also, evolutionary algorithms are less susceptible to the shape or continuity of the Pareto front, whereas these two issues are a real concern for mathematical programming techniques.

CHAPTER 5. SOLUTION METHODOLOGY ON A SERIES SYSTEM

5.1 Introduction

In Chapters 3 and 4 the basis of the GAs and Performance Based Logistics concept have been described, respectively. Multiple objective genetic algorithms (MOGA) are proposed to optimize the different conflicting objectives mentioned before. First, it randomly takes a population of chromosomes and evaluates the fitness of the results. The best solutions are retained (selection) and a new population is created (reproduction), incorporating mutation and crossover operations to gain a different set of possibilities (variation). Over many generations the population will search the space and hopefully converge on the best solution, the global optimum.

5.2 Illustrative Example: The design of a series system

An unmanned aerial vehicle that needs to be designed from a set of available components offered from different vendors. The main system consists of four major subsystems arranged in a series configuration and for each subsystem there are three available alternatives. The system has an expected design life of ten years, an annual discount rate of 12% an exponential distribution has been assumed for the data. Table 2 contains the annual component failure and repair rates and Table 3 shows each component's TCO of the alternative design.

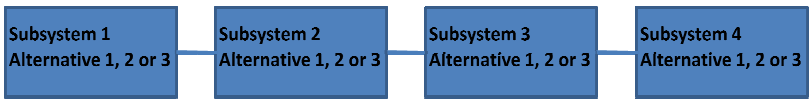


Figure 22. Series System

Table 2. Annual component failure and repair rates

	Failure rates λ_{ij}			Repair rates μ_{ij}		
	1	2	3	1	2	3
1	0.04	0.042	0.039	4	3.5	4.2
2	0.016	0.018	0.0155	5.5	4.8	6
3	0.0106	0.0112	0.011	3.8	2.9	4.5
4	0.012	0.0119	0.0108	4.4	4	3.8

Table 3. Component TCO

	Unit cost c_{ij} (dlls)			Average repair cost m_{ij} (dlls)			Operating cost o_{ij} (dlls)		
	1	2	3	1	2	3	1	2	3
1	12000	20000	25000	340	435	342	20000	25000	18000
2	46000	70000	68500	255	260	280	70000	68500	71000
3	34000	40000	41500	400	415	372	40000	41500	45000
4	52000	56000	55000	380	410	380	56000	55000	45000

5.3 Problem Description

During the design stage of a product development, the designers have the difficult task of choosing different architectures and product alternatives that would result in a maximum benefit to the user. The challenging task during the design stage is to simultaneously optimize different conflicting objectives, such as the reliability and maintainability at a lower cost. Thus, a multiple objective optimization problem requires solving two different problems, firstly to establish a statement of the problem in a suitable mathematical form called the 'objective function' (definition of the design space) and secondly to somehow search this space to locate acceptable solutions (global optima) in terms of the 'decision variables'.

5.4 Objectives

The development of a new multiple objective evolutionary algorithm that simultaneously optimizes the different objectives in conflict is proposed. The different objectives consider in this problem are:

Objective 1. Maximize Reliability. The reliability for a system is given by:

$$\text{Max } R = [\prod_{i=1}^{n} R_i(t)]$$

Objective 2. Minimize Total Cost of Ownership (TCO). For a system with *n* subsystem the total cost of ownership is given by:

$$\text{Min } TCO = \sum_{i=1}^{n} \sum_{j=1}^{n_i} c_{ij} \times \delta_{ij} + \sum_{i=1}^{n} \sum_{j=1}^{n_i} (\lambda_{ij}\, m_{ij} + o_{ij})\delta_{ij} \times K_L$$

Where: $$K_L = \frac{1-(1+r)^{-L}}{r}$$

Objective 3. Minimize Mean Time to Repair (MTTR):

$$\text{Min } MTTR = \sum_{i=1}^{n} \sum_{j=1}^{n_i} \frac{\lambda_{ij}\mu_{ij}\delta_{ij}}{\lambda}$$

Where λ is the system failure rate and is given by:

$$\lambda = \sum_{i=1}^{n} \sum_{j=i}^{n_i} \lambda_{ij}\delta_{ij}$$

The notation used as follows:

- n = Is the total number of subsystems.
- n_i = Number of design alternatives available for subsystem *i*.
- L = Design Life.
- r = Discount rate per year.
- c_{ij} = Unit cost of component type *j* in subsystem *i.*
- o_{ij} = Operating cost of component type *j* in subsystem *i.*

- λ_{ij} = Annual failure rate of component type *j* in subsystem *i*.
- μ_{ij} = Annual repair rate of component type *j* in subsystem *i.*
- m_{ij} = Average repair cost of component type *j* in subsystem *i*.
- δ_{ij} = Variable associated of component type *j* in subsystem *i*.

5.5 Solution Methodology

The description of how this multiple objective problem was solved using GAs is presented next and is shown in figure 23.

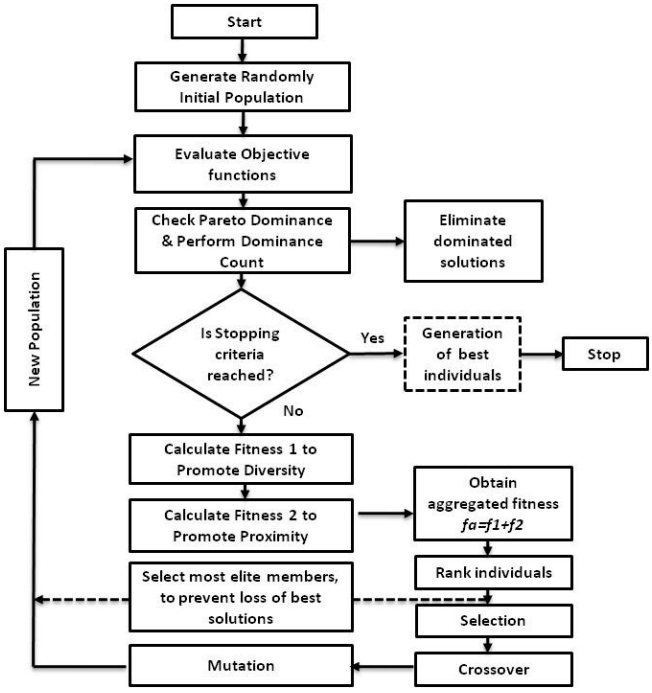


Figure 23. Multiple Objective Genetic Algorithms flowchart

1. *Encoding*. This is the first step in order to start solving a problem using GAs. It refers to how the chromosomes are presented in the problem. The type of encoding depends on the problem. In this case a value encoding is used. The use of binary encoding for this type of problems would be very difficult or would not be sufficient. In

value encoding, every chromosome is a string of some values. Values can be anything connected to the problem.

In GAs each bit or block is called gene and represents an alternative for the subsystem. A group of genes constitute a chromosome, and this is considered as a possible solution for the problem.

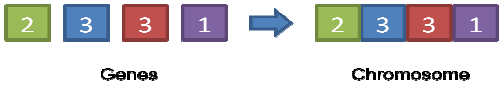


Figure 24. Value Encoding

For this specific chromosome with four subsystems, it is assumed that the subsystem 1 is considering option 1, for subsystem 2 the alternative 2 is considered, for subsystem 3 alternative 2 is considered, and finally, for subsystem 4 the alternative 3 is considered.

2. *Initial Population.* This is determined randomly by selecting *N* possible solutions. In general, the minimum effective population-size grows with problem-size. For illustration purposes, a small population is considered with *N*=10.

Table 4. Initial population

Individual	*Subsystem 1*	*Subsystem 2*	*Subsystem 3*	*Subsystem 4*
1	1	2	1	1
2	1	2	3	2
3	1	3	1	2
4	1	3	1	3
5	1	3	2	2
6	2	1	1	1
7	2	1	1	2
8	2	2	1	1
9	2	3	2	2
10	2	3	2	3

3. *Evaluation of the objective function*. The evaluation of the objectives for the three objectives functions for each chromosome in the population is shown below:

Table 5. Objective function evaluation for the initial population

Individual	*Reliability*	*TCO*	*MTTR*
1	0.446641	1210493	4.211911
2	0.445303	1244594	4.245983
3	0.458406	1221469	4.370256
4	0.463476	1163966	4.347464
5	0.455664	1235944	4.237659
6	0.446641	1231224	4.070471
7	0.447088	1229574	4.010932
8	0.437797	1246749	3.952542
9	0.446641	1272200	3.971216
10	0.451581	1214697	3.943648

4. *Pareto Dominance*. The next step is the evaluation of the objective functions with the Pareto dominance principle. The solutions that will dominate among the others will continue, the others will be eliminated.

In order to start with this process, it is necessary to perform the dominance count of how many individuals are dominating the others.

Table 6. Dominance count in the first population

Individual	*Reliability*	*TCO*	*MTTR*	*Dominance count*
1	0.446641	1210493	4.211911	2
2	0.445303	1244594	4.245983	0
4	0.463476	1163966	4.347464	0
6	0.446641	1231224	4.070471	1
7	0.447088	1229574	4.010932	1
8	0.437797	1246749	3.952542	1
9	0.446641	1272200	3.971216	0
10	0.451581	1214697	3.943648	2

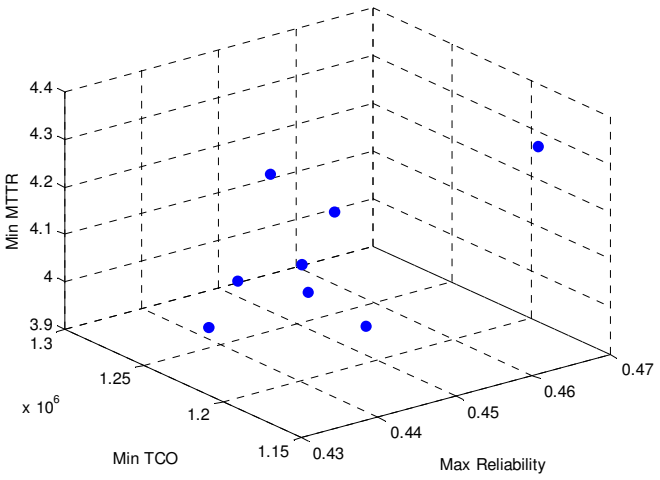


Figure 25. Pareto Front for the first generation

The maximum dominance count in this case is 2 from the individuals 1 and 10. The dominated individuals will be removed from the population and the nondominated solutions continue in order to evaluate their fitness. Even when individuals 2, 4 and 9 do not dominate any solution they still are a nondominated because they are not nominated by any other.

5. *Fitness evaluation*. In order to perform this part of the algorithm, two different fitness metrics were considered as proposed in Taboada & Coit, 2008:

• *Distance based:* This is the first fitness consider, it helps to achieve population diversity. The basis of this method works by assigning the highest fitness for those individuals that are far away from the other individuals in the Pareto front. It means that the distance d_i from individual i to the rest of the individuals are evaluated and then the resulting values are summed up. The higher the fitness value is, the more the possibilities to be selected for the next generation. This is very important in order to

prevent the early convergence and to guarantee that the solution space is totally considered.

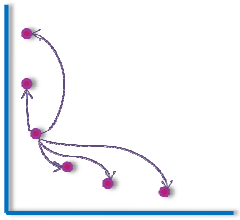


Figure 26. Diversity

Then standardization of the objectives is needed in order to have comparable units.

Table 7. Standardized values for the nondominated solutions

Individual	Max Reliability	Min TCO	Min MTTR	Dominance count
1	0.6555954	0.429874	0.66432	2
2	0.707696	0.744942	0.748695	0
4	0	0	1	0
6	0.6555954	0.621413	0.314061	1
7	0.6381934	0.606168	0.16662	1
8	1	0.764852	0.022025	1
9	0.6555954	1	0.068269	0
10	0.4632157	0.468716	0	2

The next step is the calculation of the Euclidian distance from each individual to the others, at the end, the distances will be added up and the maximum and minimum values will be determined.

Table 8. Euclidian distance for the different solutions

Individual	1	2	4	6	7	8	9	10
1	0	0.33030	0.85281	0.39921	0.52829	0.80210	0.82482	0.69270
2	0.33030	0	1.05779	0.45484	0.60241	0.78351	0.72853	0.83463
4	0.85281	1.05779	0	1.13423	1.21213	1.59419	1.51589	1.19761
6	0.39921	0.45484	1.13423	0	0.14924	0.47379	0.45138	0.39870
7	0.52829	0.60241	1.21213	0.14924	0	0.42070	0.40630	0.27798
8	0.80210	0.78351	1.59419	0.47379	0.42070	0	0.41958	0.61345
9	0.82482	0.72853	1.51589	0.45138	0.40630	0.41958	0	0.56915
10	0.69270	0.83463	1.19761	0.39870	0.27798	0.61345	0.56915	0
Sum	4.43023	4.79202	8.56464	3.46139	3.59705	5.10732	4.91564	4.58423

The maximum and minimum values are found and those will be the ones used to form the intervals. The intervals will give the fitness value to each individual; the number of intervals will be established by the designer, in this case 5 intervals are shown.

Table 9. Fitness value 1 for the nondominated solutions

Fitness 1	*Intervals*	*Individual*
1	3.46139 ≤ Sum ≤ 4.48204	6 and 7
2	4.48204 ≤ Sum ≤ 5.50269	1, 2, 8, 9 and 10
3	5.50269 ≤ Sum ≤ 6.52334	0
4	6.52334 ≤ Sum ≤ 7.54399	0
5	7.54399 ≤ Sum ≤ 8.56464	4

And the values for the fitness 1 are shown below.

Table 10. Fitness 1 "Population Diversity"

Individual	*Reliability*	*TCO*	*MTTR*	*Fitness value 1*
1	0.65560	0.42987	0.66432	2
2	0.70770	0.74494	0.74869	2
4	0.00000	0.00000	1.00000	5
6	0.65560	0.62141	0.31406	1
7	0.63819	0.60617	0.16662	1
8	1.00000	0.76485	0.02202	2
9	0.65560	1.00000	0.06827	2
10	0.46322	0.46872	0.00000	2

• *Dominance count based*. This is the second fitness metric used to encourage proximity to the Pareto front. This is the most common criterion used; working by itself it is not sufficient so for most of the time it needs to be combined with some other criterion. This method works by using the standardized nondominated solutions and their dominance count.

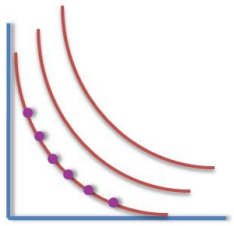


Figure 27. Proximity

Table 11. Standardized for the nondominated solutions

Individual	Reliability	TCO	MTTR	Dominance count
1	0.655595	0.429874	0.66432	2
2	0.707696	0.744942	0.748695	0
4	0	0	1	0
6	0.655595	0.621413	0.314061	1
7	0.638193	0.606168	0.16662	1
8	1	0.764852	0.022025	1
9	0.655595	1	0.068269	0
10	0.463216	0.468716	0	2

The previous table shows that the maximum dominance value is 2 and the minimum is 0. Similar as in previous fitness metric intervals will be considered.

Table 12. Fitness value 2 for the nondominated solutions

Fitness 2	Intervals	Individual
1	0 ≤ Dominance count ≤ .4	2, 4 and 9
2	.4 ≤ Dominance count ≤ .8	0
3	.8 ≤ Dominance count ≤ 1.2	6, 7 and 8
4	1.2 ≤ Dominance count ≤ 1.6	0
5	1.6 ≤ Dominance count ≤ 2.0	1 and 10

Once the fitness values are obtained for both methods, one to achieve the population diversity and the other to find out which individuals are the most dominating in the population, an aggregated fitness value will be used for ranking the solutions.

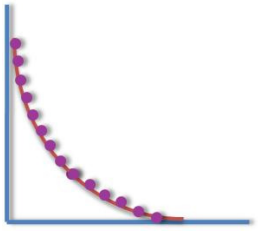


Figure 28. Aggregation of fitness

The following table contains the values for both fitness metrics and the aggregation of them.

Table 13. Aggregated fitness values

Individual	*Reliability*	*TCO*	*MTTR*	*Dominance count*	*Fitness value 1*	*Fitness value 2*	*Aggregated Fitness value*
1	0.65560	0.42987	0.66432	2	2	5	7
2	0.70770	0.74494	0.74869	0	2	1	3
4	0.00000	0.00000	1.00000	0	5	1	6
6	0.65560	0.62141	0.31406	1	1	3	4
7	0.63819	0.60617	0.16662	1	1	3	4
8	1.00000	0.76485	0.02202	1	2	3	5
9	0.65560	1.00000	0.06827	0	2	1	3
10	0.46322	0.46872	0.00000	2	2	5	7

The highest fitness values are for those solutions that are more dominating and have more possibilities to go to the next generation, the ones with the lowest fitness values are the least desirable individuals.

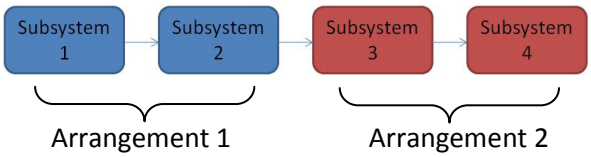
6. *Selection.* The selection rank method is used with a 25% of elitism and the best solutions go directly to the next generation in order to prevent losing the best solutions. In this particular case after ranking all the individuals according to their fitness value, these are the selected solutions:

Table 14. Rank Selection

Individual	Reliability	TCO	MTTR	Dominance count	Fitness value 1	Fitness value 2	Aggregated Fitness value
1	0.65560	0.42987	0.66432	2	2	5	7
10	0.46322	0.46872	0.00000	2	2	5	7
4	0.00000	0.00000	1.00000	0	5	1	6
8	1.00000	0.76485	0.02202	1	2	3	5
6	0.65560	0.62141	0.31406	1	1	3	4
7	0.63819	0.60617	0.16662	1	1	3	4
2	0.70770	0.74494	0.74869	0	2	1	3
9	0.65560	1.00000	0.06827	0	2	1	3

The rest of the individuals will need a recombination of their genes and they go directly to the crossover step.

7. *Crossover.* Once the selection of the best individuals is done, the creation of new individuals is necessary by recombining their genes in order to create new offspring; this method in Genetic Algorithms is known as crossover. This is one of the factors that distinguishes GAs from other algorithms. In this case, the subsystem rotation crossover is perform due to its efficiency, where multi parent recombination is allowed and the creation of a large number of children gives the opportunity of the evaluation of more individuals, and the selection of the best ones. To mimic the stochastic nature of evolution, a crossover probability is associated with this operator, by using a probability of 75% of crossover and 2 arrangements of 2 subsystems each.



8. *Mutation*. When the crossover is done, the chromosomes are copied exactly and with a probability of 1% they will be mutated. The mutation is considered as a random

alteration of a gene in a chromosome by selecting a random point in the chromosome and substituting the value found at that point with another value that will be randomly selected.

9. *New population.* The new population will be formed by the chromosomes selected by elitism and the ones obtained from the crossover and mutation steps.

10. *Stopping criteria*. After running the program with a selected number of generation, the algorithm stops.

The Pareto optimal set obtained at the end of 5 generations is shown in figure 29. From here, the decision maker needs to select one solution for system implementation. This is a small example and was used only with illustration purposes. Chapter 6 presents a larger example.

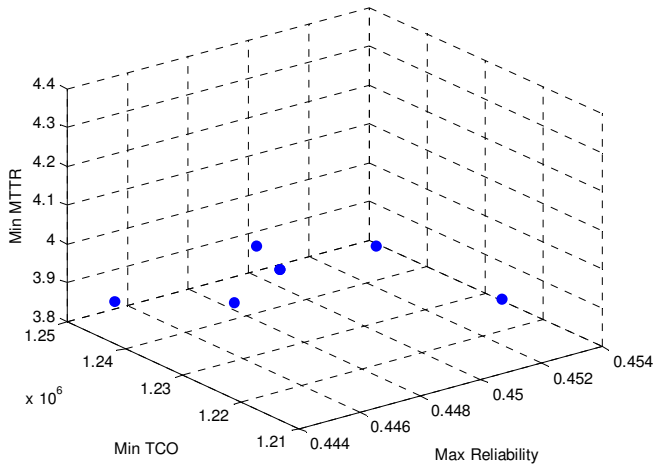


Figure 29. Pareto front for the last generation

CHAPTER 6. SOLUTION METHODOLOGY ON A SERIES-PARALLEL EXAMPLE

This chapter presents two different examples to illustrate the performance of the developed algorithm. A multiple objective optimization problem is formulated considering the optimization of three objective functions that are considered relevant to the PBL area. The problem focuses mostly on the design stage.

This section describes how the proposed multiple objective evolutionary algorithm works. The main objective of the algorithms is to find a set of Pareto-optimal solutions that simultaneously optimize all the conflicting objectives in a series-parallel system. The objective functions that need to be simultaneously optimized are the maximization of system reliability, the minimization of the total cost of ownership and the minimization of the mean time to repair.

6.1 Example 1

For the first example, a manufacturing company specialized in the process of aircraft's harnesses is considered. The system consists of four main subsystems that performed the following functions: routing, stripping, crimping and insertion, respectively. These four subsystems are arranged in a series-parallel system as can be seen in figure 30, for each subsystem there are different components/pieces that need to be bought in order to complete the process, and for each one there are different available alternatives to choose. From the system has an expected design life of ten years for which an annual discount rate of 12% has been assumed. Failure times and repair times are assumed to be exponentially distributed random variables. Table 15

contains the annual component failure and repair rates and Table 16 shows each component's TCO of the alternative design.

The following list shows the parameters and some of the computer characteristics used to run the developed algorithm:

- 4 Subsystems
- 6 Components
- 3 Component Alternative
- 25% Elitism
- 75% Crossover
- 1% Mutation
- 20 Population size
- 50 Generations
- Computer
  - Core 2 duo E6550
  - 2GB RAM
- MATLAB
- Time consuming 2 minutes

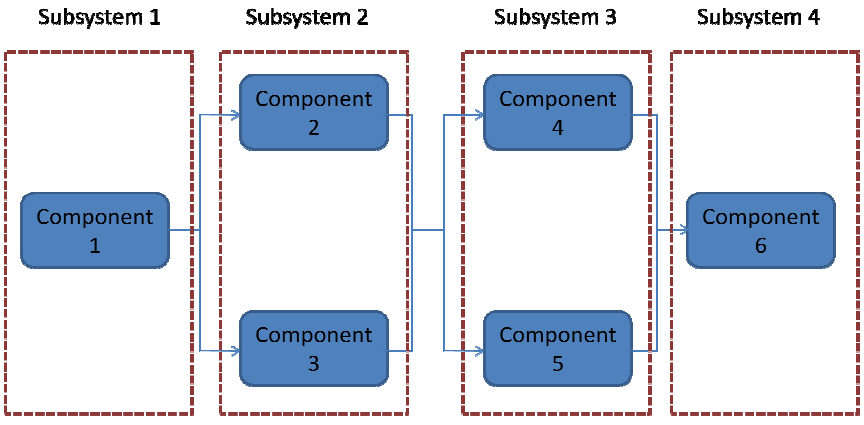


Figure 30. Series-Parallel system considered Example 1

Table 15. Component failure and repair rates for Example 1

	Failure rate λ			Repair Rate μ		
	1	2	3	1	2	3
1	0.04	0.042	0.039	4	3.5	4.2
2	0.016	0.018	0.0155	5.5	4.8	6
3	0.0106	0.0112	0.011	3.8	2.9	4.5
4	0.012	0.0119	0.0108	4.4	4	3.8
5	0.03	0.032	0.028	4.2	4.5	4.6
6	0.021	0.023	0.025	3.8	3.5	4.2

Table 16. Component's TCO for Example 1

	Unit cost c (dlls)			Average repair cost m (dlls)			Operating cost o (dlls)		
	1	2	3	1	2	3	1	2	3
1	12000	20000	25000	340	435	342	20000	25000	18000
2	46000	70000	68500	255	260	280	70000	68500	71000
3	34000	40000	41500	400	415	372	40000	41500	45000
4	52000	56000	55000	380	410	380	56000	55000	45000
5	24000	28000	35000	368	387	465	31000	35000	38800
6	29000	30500	31000	398	421	423	23700	26890	19800

After running the algorithm for 50 generations, the nondominated solutions are shown in figure 31.

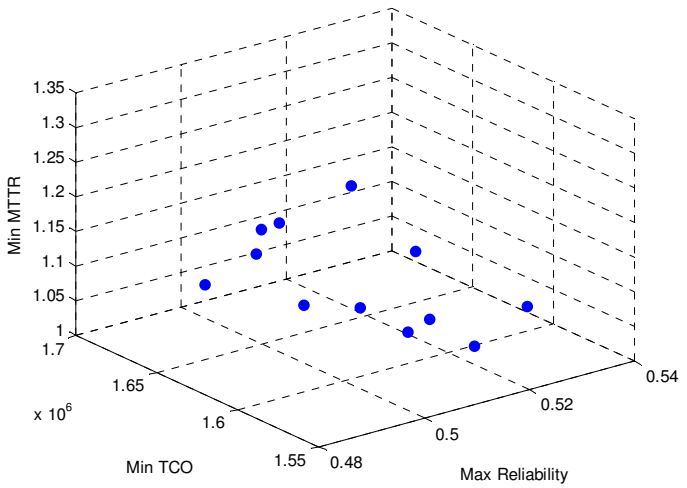


Figure 31. Pareto-optimal set of Example 1 obtained after 50 generations

6.2 Example 2

This example presents a larger and more complex system integrated by five subsystems arranged in a series-parallel configuration. For each subsystem there are different available components to build the system. These five subsystems are arranged in a series-parallel system as can be seen in figure 32. For each component there are five different available vendors. The system has an expected design life of ten years for which an annual discount rate of 12% has been assumed. Failure times and repair times are assumed to be exponentially distributed random variables. Table 17 contains the annual component failure and repair rates and Tables 18 and 19 shows each component's TCO of the alternative design.

Parameters:

- 5 Subsystems
- 12 Components
- 5 Component Alternative
- 25% Elitism
- 75% Crossover
- 1% Mutation
- 100 Population size
- 50 Generations
- Computer
  - Core 2 duo E6550
  - 2GB RAM
- MATLAB
- Time consuming 25 minutes

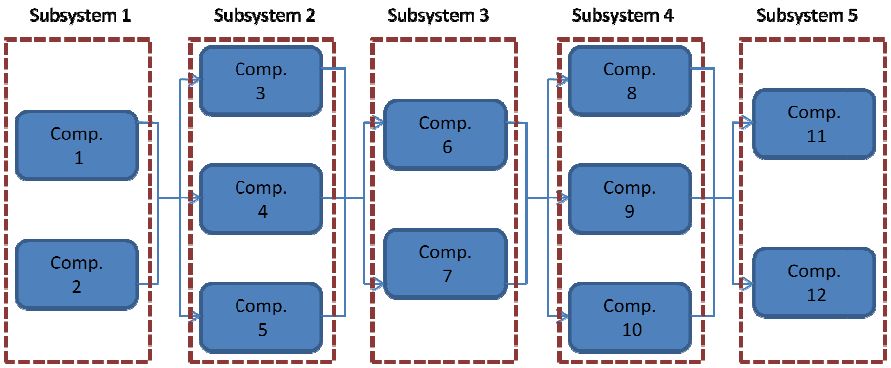


Figure 32. Series-Parallel system considered Example 2

Table 17. Component failure and repair rates for Example 2

	Failure Rate λ					***Repair Rate μ***				
	1	***2***	***3***	***4***	***5***	***1***	***2***	***3***	***4***	***5***
1	0.004	0.0042	0.0039	0.0041	0.0037	4	3.5	4.2	4.1	3.88
2	0.0016	0.0018	0.00155	0.0012	0.0017	5.5	4.8	6	4.9	5.1
3	0.00106	0.00112	0.0011	0.00101	0.00115	3.8	2.9	4.5	4.2	3.5
4	0.0012	0.00119	0.00108	0.00109	0.00125	4.4	4	3.8	3.9	4.5
5	0.03	0.0032	0.0028	0.0033	0.00029	4.2	4.5	4.6	4.9	5.1
6	0.0021	0.0023	0.0025	0.0019	0.002	3.8	3.5	4.2	4.6	4.3
7	0.0025	0.0023	0.0027	0.002	0.0029	5.6	5.2	5.8	6.1	6.2
8	0.00134	0.00143	0.00123	0.00154	0.00187	4.6	5.8	4.9	5.1	5.4
9	0.00254	0.00276	0.00198	0.00265	0.00221	3.5	3.9	3.9	3.4	3.6
10	0.00301	0.00324	0.00328	0.00298	0.00287	4.6	4.8	4.9	5.2	5.5
11	0.0045	0.0052	0.0049	0.006	0.0036	4	4.5	4.7	9.1	5.6
12	0.0019	0.0068	0.00195	0.007	0.0027	3.5	6.8	6	4.7	5.8

Table 18. Component's TCO for Example 2

	Unit Cost c (dlls)					***Average Repair Cost m (dlls)***				
	1	***2***	***3***	***4***	***5***	***1***	***2***	***3***	***4***	***5***
1	12000	20000	25000	18500	21670	340	435	342	345	365
2	46000	70000	68500	55700	48900	255	260	280	435	321
3	34000	40000	41500	47000	37000	400	415	372	487	334
4	52000	56000	55000	49000	51000	380	410	380	412	354
5	24000	28000	35000	19000	31000	368	387	465	444	476
6	29000	30500	31000	27000	25000	398	421	423	465	411
7	32000	36000	42000	47500	34000	362	498	452	347	502
8	28900	34600	26300	54280	19800	314	387	367	409	508
9	43000	32500	27980	61900	22300	375	428	312	401	501
10	38600	38000	45300	35600	43280	356	487	333	444	530
11	26000	57000	27000	33600	54270	370	655	342	345	555
12	36000	27000	57300	77400	35200	225	230	223	765	151

Table 19. Component's TCO continuation for Example 2

	Operating Cost o (dlls)				
	1	***2***	***3***	***4***	***5***
1	20000	25000	18000	29000	31000
2	70000	68500	71000	58900	72900
3	40000	41500	45000	49000	39000
4	56000	55000	45000	42000	41000
5	31000	35000	38800	45000	48000
6	23700	26890	19800	27000	36700
7	67000	56000	45000	49800	52000
8	76000	79000	59000	52000	63330
9	45600	56000	43000	35789	45876
10	67900	53000	64000	49000	54000
11	36100	25000	18000	45600	31000
12	43800	55700	71000	67500	72900

After running the algorithm for 50 generations, the nondominated solutions are shown in the Pareto set optimal solutions, as shown in the following figures.

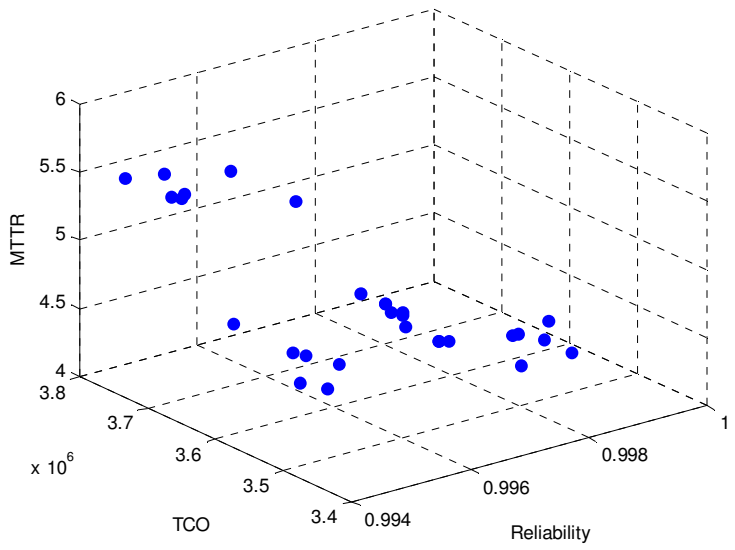


Figure 33. Pareto-optimal set of Example 2 obtained after 50 generations

All the solutions that are shown in the Pareto front are nondominated solutions, in the sense that it is not possible to get better values of any objective with the deterioration of the others; all the different combinations have the best fitness values.

The analysis for these solutions is a difficult task and requires further examination because Pareto-optimal sets can be extremely large or even contain an infinite number of solutions (Taboada & Coit, 2006). The decision maker will be in charge of performing several evaluations in order to select the one that satisfy their needs according with their requirements, desires or wishes.

There are some techniques that have been developed in order to identify the best solution from the Pareto-optimal set. Multiple objective optimization problems can be applied by combining all the objectives into a single objective function and assigned a weight to each objective or by the study of Pareto-optimal set.

The decision maker could have two different methods in order to analyze the Pareto optimal set. The first one is similar to the weighed sum method except that specific numerical weights or penalties are not required. The objectives are ranked according with their importance to the decision maker and combined their weights into a single objective function by using random weights. The best solutions are recorded and the process is repeated for a specific number of generations. The final result obtained is a set of solutions that most often provided the optimal combined objective function (Taboada & Coit).

The second approach is a clustering technique using data mining. This analysis makes it possible to look at properties of whole clusters instead of individual objects (Taboada & Coit, 2006). It consists in grouping similar solutions, and since the solutions in the cluster are similar one another, now the decision maker has fewer solutions to choose from.

In this analysis the selection of one of the solutions is done by normalizing all the objectives. The one that is the closest to the ideal point would be selected.

Ideal point:

- Reliability = 1
- TCO = 0
- MTTR = 0

Closest to ideal point:

- Reliability = .9978
- TCO = $ 3,487,122.00
- MTTR = 4.3533 hrs.

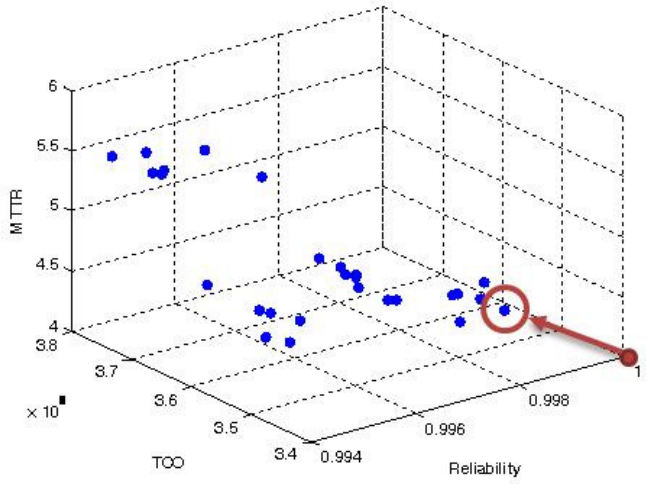


Figure 34. Selected solution

REFERENCES

1. Abraham A., Jain L., and Goldberg R., 2005, “Evolutionary Multiobjective Optimization: Theoretical Advances and Applications” Springer-Verlag London, 302pp.

2. Ajith A., Jain L., and Goldberg R., 2005, Evolutionary multiobjective optimization: Theorical Advances and Applications, Published by Springer, 2005.

3. Ali M., and Tunali S., 2007, “Joint optimization of spare parts inventory and maintenance polices using genetic algorithms,” International Journal of Advance Technology (34), 594-604.

4. Azaron A., Perkgoz C., and Katagiri H., 2009, “Multi-objective reliability optimization for dissimilar unit cold standby systems using a genetic algorithm,” Computers & Operation Research (36), 1562-1571.

5. Beggs J., Kime T., Orbacks D., and Jones M., 2005, “Performance Based Logistics: Considerations for Commercial Participants,” Booz Allen Hamilton Publications.

6. Berkowitz D., Gupta J., Simpson J., and McWilliams J., 2005, “Defining and Implementing Performance-Based Logistics in government,” Defense AR Journal, 254-267.

7. Bertsimas D., and Tsitsiklis J., 1993, “Simulated Annealing” Statistical Science, (8) 10-15.

8. Calabria R., Pulcini G., and Rapone M., 1995, “Optimal Reliability and maintainability allocation in manufacturing systems,” Quality and Reliability Engineering International, Vol. 11, 183-188. explanation

9. Cheng F., Fellow, ASCE, and Li D., 1997, “Multiobjective Optimization Design with Pareto Genetic Algorithm,” Journal of Structureal Engineering, 1252-1261.

10. Chiu C., Hsu C., and Yeh Y., 2006, “A genetic algorithm for reliability oriented task assignment with k duplications in distributed systems,” IEEE Transactions on Reliability, Vol. 55, 105-117.

11. Coit W., and Smith A., 1996, “Reliability Optimization of Series-Parallel Systems Using a Genetic Algorithm,” IEEE Transactions on Reliability, Vol. 55, 254-266.

12. Dorigo M., and Stutzle T., 2004, “Ant Colony Optimization,” Massachusetts Institute of Technology, 319pp.

13. Dorigo M., Di Caro G., and Sampels M., 2002, “Ant Algorithms: Third International Workshop, Ants 2002, Brussels, Belgium, September 12-14, 2002. Proceedings,” Springer-Verlag 305pp.

14. Duarte B., 2001, “The expected utility theory applied to an industrial decision problem-what technological alternative to implement to treat industrial solid residuals,” Computers & Operations Research (21) 357-380.

15. Dumma D. and Krieg K., 2005, “DoD guide for achieving Reliability, Availability, and Maintainability,” Department of Defense, United States of America, 266pp.

16. Ebeling C., 1997, “An Introduction to Reliability and Maintainability Engineering,” University of Dayton, Waveland Press, Inc., 504pp.

17. Figueira J., Greco S., and Ehrgott M., 2005, “Multiple criteria decision analysis: state of the art surveys,” Springer Science Bussines Media, 1045pp.

18. Fonseca C., and Fleming P., 1993, “Genetic Algorithms for Multiobjective Optimization: Formulation, Discussion and Generalization,” Dept. Automatic Control and Systems Engineering, University of Sheffield, Sheffield, U.K.

19. Gansler J., and Lucyshyn W., 2006, “Evaluation of Performance Based Logistics,” Center for Public and Private Enterprise, University of Maryland.

20. Giannotti E., Negron R., Antzoulis T., 2006, “An integrated information system approach for Performance Based Logistics,” IEEE, 336-340.

21. Glover F., 1989, “Tabu Search-Part I” Operations Research Society of America.

22. Glover F., 1990, “Tabu Search: A Tutorial,” Institute of Management Sciences, 74-94.

23. Glover F., 1990, “Tabu Search-Part II” Operations Research Society of America.

24. Glover F., and Laguna M., 1997, “Tabu Search,” Kluwer Academic Publishers, 404pp.

25. Goure D., 2009, “Performance Based Logistics: A primer for the new administration,” Lexington Institute, Arlington, Virginia.

26. Gutierrez C. and Briones A., 2009, “Pareto front of ideal Petlyuk sequences using multiobjective genetic algorithm with constrains” Computer and Chemical Engineering (33) 454-464.

27. Hou W., and Okogbaa G., 2005, “Reliability and Availability cost design tradeoffs for HA system,” IEEE Xplore, 433-438.

28. Hu X., and Eberhart R., 2002, “Adaptive Particle Swarm Optimization: Detection and Response to Dynamic Systems” IEEE 1666-1670.

29. Kalyanmoy D., 2001, “Multi-objective optimization using evolutionary algorithms,” Wiley, 518pp.

30. Keating S. and Huff K., 2005, “Managing risk in the new supply chain,” IEE Engineering Management.

31. Keeney R., and Raïffa H., 1993, “Decisions with multiple objectives: preferences and value tradeoffs,” Cambridge University Press, 592pp.

32. Kennedy, J., Eberhart, R. C., and Shi, Y., 2001, “Swarm Intelligence,” Academics Press, 512pp.

33. Kim Y., and Weck O., 2004, “Adaptive Weighted Sum Method for Multiobjective Optimization,” American Institute of Aeronautics and Astronautics, Inc.13pp.

34. Konak A., Coit D. and Smith A., 2006, “Multi-objective optimization using genetic algorithms: A tutorial,” Reliability Engineering and System Safety (91)992–1007.

35. Laarhoven P., and Aarts E., 1987, “Simulated Annealing: Theory and Applications” Reidel Publishing Company, 204pp.

36. Man K., Tang K., and Kwong S., 2001, “Genetic Algorithms,” Springer-Verlag, 348pp.

37. Martinez M., Herrero J., and Sanchis J. 2009 “Applied Pareto multi-objective optimization by schocastic solvers” Engineering Applications of Artificial Intelligence (22) 455-465.

38. Miettinen K., 1998, “Nonlinear multiobjective optimization,” Kluwer Academic Publishers, 320pp.

39. Mitchell M., 1996, “An introduction to genetic algorithms,” MIT Press, 221pp.

40. Nowicki D., Ramirez J., and Verma D., 2007, “A goal programming model for optimizing Reliability, Maintainability and Supportability under Performance Based Logistics,” International Journal of Reliability, Quality and Safety Engineering, Vol. 14, 251-261.

41. Onwubolu G., and Babu B., 2004, “New optimization techniques in engineering” Springer-Verlag 712pp.

42. Painton L., and Campbell J., 1995, “Genetic algorithm in optimization of System Reliability,” IEEE Transactions on Reliability, Vol. 44, 172-178.

43. Price R., 1991, “Weapon system reliability and maintainability: Increasing their visibility with decision makers,” 1991 Procedings Annual Reliability and Maintainability Symposium, 82-84.

44. Reeves C., and Rowe J., 2003, “Genetic Algorithms: Principles and Perspectives: a guide to GA theory,” Kluwer Academics Publisher, 332pp.

45. Salamon P., Sibani P., and Frost R., 2002, “Facts, conjectures, and improvements for simulated annealing,” Society for Industrial and Applied Mathematics, 136pp.

46. Schniederjans M., 1995, “Goal programming: methodology and applications,” Springer-Verlag New York, 236pp.

47. Smith T., 2004, “Reliability Growth Planning Under Performance Based Logistics,” IEEE Xplore, 418-423.

48. Taboada, H. & Coit, D., “Post-Pareto Optimality Analysis to Efficiently Identify Promising Solutions for Multi-Objective Problems.”

49. Taboada H., and Coit D., 2007, “Data clustering of solutions for multiple objective system reliability optimization problems,” Quality Technology & Quantitative Management Journal (2) 35-54.

50. Taboada H., Espiritu J., and Coit D., 2008, “MOMS-GA Multi-Objective Multi-State Genetic Algorithm for System Reliability Optimization Design Problems,” IEEE Transactions on Reliability, Vol. 57, 182-191.

51. Taboada, H. & Coit, D., 2006. “Data Mining Techniques to Facilitate the Analysis of the Pareto-Optimal Set for Multiple Objective Problems.” In Proceedings of the Industrial Engineering Research Conference (IERC), Orlando, Florida, May 2006.

52. Taboada, H. & Coit, D., 2009, “MOEA-DAP: A new Multiple Objective Evolutionary Algorithm for solving Design Allocation Problems” Reliability Engineering & System Safety (under review).

53. Vitasek K. and Geary S., 2008, “Performance Based Logistics: Redefines Department of Defense Procurement,” WorldTrade, 62-65.

54. Whitley D., 1994, “A Genetic Algorithm Tutorial,” Statistics and Computing (4): 65-85.

55. Yang C., and Simon D., 2005, “A New Particle Swarm Optimization Technique,” Proceedings of the 18th International Conference on Systems Engineering, 164-169.

CURRICULUM VITA

Delia Villanueva was born in Chihuahua, Chih., Mexico. The first of two daughters of Jaime Villanueva and Delia Jaquez. She received her bachelor degree in the Chihuahua Institute of Technology in spring 2007. Then she started her professional career as a Process engineer at an aircraft harness manufacturing company called Labinal in Chihuahua City. Later, she entered The University of Texas at El Paso in Spring 2008 with the Chihuahua government Scholarship. While pursuing a Master's Degree in Industrial Engineering, she worked with Dr. Heidi Taboada as her Teacher Assistant and Research Assistant. Professionally she had the opportunity to present her research in the 2009 UTEP SACNAS Expo Research, in El Paso Texas. Also, she presented at the 2009 INFORMS Western Regional Conference in Tempe, Arizona and finally, at the 2009 INFORMS Annual Conference in San Diego, California.