A decision support model for process planning in remanufacturing using fuzzy set theory and genetic algorithm

Juan C. Martinez
University of Texas-Pan American

A DECISION SUPPORT MODEL FOR PROCESS PLANNING IN REMANUFACTURING USING FUZZY SET THEORY AND GENETIC ALGORITHM

A Thesis

by

JUAN C. MARTINEZ

Submitted to the Graduate School of
The University of Texas-Pan American
In partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

August 2014

Major Subject: Manufacturing Engineering

A DECISION SUPPORT MODEL FOR PROCESS PLANNING IN REMANUFACTURING USING FUZZY SET THEORY AND GENETIC ALGORITHM

A Thesis
by
JUAN C. MARTINEZ

August 2014

ABSTRACT

Martinez, Juan C., A Decision Support Model for Process Planning in Remanufacturing using Fuzzy Set Theory and Genetic Algorithm. Master of Science (MS), August, 2014, 90 pp., 11 tables, 20 figures, references, 31 titles

The environmental awareness and resources preservation has boosted the use of remanufacturing processes. Many important studies have been made in this area, but there are still pitfalls and opportunities that have not been explored yet. Specifically, the cleaning processes play a critical role in remanufacturing. It is among the top of most costly processes and it plays a key step in remanufacturing because delivers a free of contaminant product ready for reprocessing. Understanding and selecting the best processes to meet the need and goals of remanufacturers are important for effective and efficient remanufacturing. Due to variables and high uncertainty associated with cleaning processes, there is a need of both knowledge and application of remanufacturing process planning. Aiming on this need, a decision making tool based on fuzzy sets and genetic algorithm for process planning is proposed. It helps the remanufacturer to select the best cleaning processes to achieve its objective.

DEDICATION

The completion of my master's studies was possible because of the support of my family and friends. To my Father, the one that showed me perseverance, hard work and that it does not matter what happens in our life, there is always a path to success. To my Mother, the one that showed me to be organized, focused and always do the best to achieve any goal. To my brothers, that even though we are geographically separated, we all are doing whatever it takes to achieve success. Finally, to my friends for their support and kindness.

ACKNOWLEDGEMENTS

I will always be very grateful to my mentor and advisor, Dr. Jianzhi Li, for believing in me and pushing me to be a better person through this whole process. He helped all the way from choosing the topic, understanding the current knowledge and identifying critical opportunities that will make this world a better one. My thanks goes also to my committee members, Dr. Miguel Gonzalez, Dr. Douglas Timmer and Dr. Hiram Moya for their inputs and to ensure that my work really has a high quality standard.

A special recognition goes to Julio Oyervides and Raúl Estrada for helping me with the Matlab® programming. With them, all the programming work was possible and very smooth. Also, I want to thank my colleagues and laboratory partners Hugo Montiel, Cynthia Barros and Yingbin Hu for making the work more amenable.

TABLE OF CONTENTS

Page

Page

LIST OF TABLES

LIST OF FIGURES

Page

CHAPTER I

INTRODUCTION

This chapter is dedicated to show the overview of the investigation. The topics that includes are: 1) problem review, 2) study purpose and research questions, 3) thesis organization.

Problem Review

Remanufacturing differs from normal manufacturing in many perspectives. There is a higher level of uncertainties in supply and demand of remanufactured parts. There are others factors that does not exist in a manufacturing process such as: condition of the product, level of contamination, wear, debris, oxidation, etc. This is attributed to the issues that remanufacturing faces when transforms a used product to a “like new” condition. Generally, the used product needs to go through a series of steps such as: inspection, disassembly, cleaning, re-processing and testing to achieve a “like new” product. Among these processes, studies had proved that the cleaning process is one of the most important steps in a remanufacturing.

There are many cleaning methods used in the remanufacturing process. Due to the uncertainty of the product condition and product usage condition before the moment of reception, the same cleaning method cannot be used all the time. Many evaluation of the cleaning methods need to be perform in order to decide which one should be used to achieve a desired cleaning level demanded by the remanufacturing process. Not all the cleaning processes can clean all the different contaminations at the same rate of time and cost. Some are good for some types than others. Many different cleaning processes have been created, which produce different results depending on the initial condition. Cleaning process generally fall in to following group based on the technology or clean media used: ultrasonic cleaning, abrasive cleaning, laser cleaning, thermal cleaning, chemical cleaning, etc. With all these options, the remanufacturer has to have a comprehensive understanding of the different process parameters and performance measures in order to select the best cleaning processes that meet their need. The process parameters of a cleaning process generally involve: contamination level, contamination type, cleaning system type, material type, cleaning chemicals, temperature, and process cycle time; while performance measures may include: efficiency, system energy consumption, operating cost per part, system cost and emission levels and cleaning effectiveness. Many other factors may also affect the performance and adoption of cleaning method. This includes: condition and geometry of the parts, clean media restriction, energy consumption constraint, environment impact, quantity and types of contaminants, material resistance, and etc.

Given all the issues discussed, it becomes very difficult for the remanufacturer to choose the best processes that reduce cost, reduce time or achieve maximum cleanliness level.

Study Purpose and Research Questions

The cleaning process is one with the highest cost amongst the remanufacturing processes. There is however, a lack of rigorous models for the different cleaning processes that can relate inputs conditions with the outputs such as cost, cleaning performance or energy consumption. Most information available for the cleaning processes is vague and is provided in ranges for equipment cost, energy consumption, cost per piece, etc. Also, each cleaning processes has its advantages and disadvantages in terms of cleaning capability. Some are better suitable for soft materials because the process is more delicate, but in the main time it can also only clean soft contaminants. Others may have a better cleaning effects but the piece may be harmed during the process. Among these issues, each process has to deal with hazardous wastes that may be created after cleaning the product. Also, there are many different processes to choose from that can be overwhelming for the user. With the importance of the cleaning process as an essential stage of the process planning in remanufacturing, it is crucial to develop method and tool that assist remanufacturers choose the best ones to avoid future problems. But as explained, there many variables and pitfalls that make very difficult choosing the right cleaning process.

Due to high complexity, a decision making tool that helps the remanufacturer develop process planning with their desired goal is thus proposed in this thesis work. The overall methodology is based on mathematical programming model (PM) with fuzzy sets (FS) and genetic algorithm (GA) as the solution approach. The PM is used to model the objective function with the constraints and decisions variables. The FS are used to deal with the different uncertainties and lack of "exact" information that exist in remanufacturing. FS helps to work with ranges that the remanufacturer can establish depending on their need and knowledge of the process. The GA is used to find the best possible solution (close to optimal) that meets the different constraints.

Thus, the purpose of the study is to create a decision making tool that a remanufacturer could use in their process planning to reduce process time, reduce cost or achieve a desired level of cleanliness needed to achieve the "like new" products that remanufacturing industry demands.

Thesis Organization

This thesis is divided into four chapters. Chapter one refers to the problem review and introduction to the study. Chapter two is dedicated to the literature review of the cleaning processes, surface contamination, fuzzy set theory, genetic algorithm. Chapter three covers the PM model, the fuzzy sets, membership function, fuzzy inference and the input/output of the model and the genetic algorithm. Chapter four includes a case study to validate the model along with the conclusions and future research.

CHAPTER II

LITERATURE REVIEW

This chapter is dedicated to explain all the different concepts and actual knowledge that exists in the areas of: cleaning processes, surface contaminations, fuzzy set theory and genetic algorithm. The importance of this chapter is that explains the theory and applications needed in the model for the decision making tool that is developed in this study.

Remanufacturing

Remanufacturing is a process that deliver "like – new" product that increment the life-cycle of the product. The normal life cycle has a point on which the product needs to be retired due to wear, contaminations or decreased performance. Remanufacturing delivers a product that increment its life-cycle. The remanufacturing process receives a used/retired product and put in a condition that translate in a larger life-span and better reliability (Junior and Filho, 2011). The word remanufacturing has many synonyms like rebuilding, refurbishing and overhauling. The steps are: disassembly, clean, recondition/replenish (OEM) and reassembly. Tests and inspections are done through the process to achieve the quality desired. One of the main characteristics of this process is that gives a warranty of the whole product (like-new or lifetime) (Steinhilper, 1995). The remanufacturing process has gain importance in the process because of the many opportunities that provides such as: sustainability, new challenges, job creations and affordable prices (Steinhilper, 1995). Even though remanufacturing it is being used by big manufacturer companies, there is some rejection from the end customer. A study performed in Brazil provides comparison between manufacturing and remanufacturing process. They did two decision making process to decide which type of product they used. When taking in consideration only three variables: Cost, Environmental Impact and Technical Performance, remanufacturing was the best choice (according to their case study). Surprisingly, when customer is put into the decision making equation as a decision variable, in conjunction with the other three variables, remanufacturing does not look so attractive anymore. Overall cost was one of the decision discrepancy. Also, they mention that the cost of cleaning has to be reviewed (Barba, Oliveira, Salis and Schuch, 2013).

According to WeiWei and Bin (2013) *"the availability, quality, remanufacturing cost and the remaining life of the remanufactured product will be directly influenced by various cleaning methods and the corresponding cleaning quality."* They propose a study of the cleaning technology that should be used in remanufacturing.The authors also states: "*In spite of the simplification and effectiveness of present remanufacturing cleaning process, unified standards for cleanliness judgment and the knowledge base of remanufacturing cleaning are insufficient."*

The statement is complemented with a problem that exists in cleaning efficiency due to low level of process automation. Also, that is needed a national and international standards to achieve a better level of cleaning efficiency in the remanufacturing processes (WeiWei and Bin, 2013).

In Australia there is a plan to achieve 95% of recovery potential of the parts. But in real world processes, that level is very difficult to achieve (El Halabi and Doolman, 2013). There are studies that explain that a product design should take in consideration the remanufacturing process to improve the recovery rate. If the design process takes in consideration the remanufacturing part, the organization can prepare and plan to the future in order to achieve a higher rate of recovery. This is one of the many problems and challenges that exist in remanufacturing. Five operational challenges in remanufacturing were found and should be optimized to achieve the 95% goal: 1) Supplies of End of Line Vehicles (ELV); 2) Demand for Used parts; 3) Premises; 4) Workforce; and 5) Industry Image (El Halabi and Doolman, 2013). This is another point of why remanufacturability should be taking in consideration in the design stage. A Fuzzy – QFD approach was developed to identify the better attributes that the remanufacturer needs in their processes and that should be evaluated in the design process (Yang, Ong and Nee, 2013). Take the cleaning process as an example. If a type of durable coat, special paint or additive that avoids wear is taken in consideration in the design process, the cleaning process in the remanufacturing may be cheaper to perform, because it is known beforehand that the coat needs to be removed in order to clean the surface. So, the planning process is more tangible to perform. They can run experiments to develop a cleaning process that works with a variety of conditions.

In the new challenges that exist in remanufacturing, the cleaning process has a very important role. This step delivers a product that is ready for reprocessing. The process needs to have many important quality measures that assures the overall cleaning of the product/piece.

According to a survey performed ranks the cleaning process at the top of most costly inside the remanufacturing (Hammon, 1998). It is ranked second (29%) just after part replacement (43%) given by the expert systems surveyed.

One of the challenges that exist in actual times is the lack of decision making tools that helps the remanufacturer to decide which type of method use, achieving the level of cleanliness desired at lower cost/time possible. This is gap that needs to be filled in order to contribute in the cleaning processes inside the remanufacturing, and go a step forward in this knowledge area.

Cleaning Processes

This part of the chapter summarizes the different cleaning methods that exist in current times. Ozan Yagar, in his thesis “Remanufacturing cleaning process evaluation, comparison and planning”, did a comprehensive understanding of the different processes. A summary of those processes are going to be used here for quick reference.

Immersion Cleaning

As the name refers, the piece is submerged in an aqueous media via different methods. This process with convection currents and vibrations to remove the surface contamination. The approaches to use immersion cleaning includes belt conveyors and rotary drums where the pieces are put inside the media. Mechanical agitation in the aqueous media can be used to improve results. Another alternative is to use high pressure pumps to generate a flow in the media to clean the piece (Li and Li). This process is the most suitable for removing soiling particles, films and coating, oils, soils, carbon, rust, dirt and gaskets from solid surfaces. Also, it cleans regular and complex shapes, pieces with holes and parts hard to reach. For medium and heavy duty immersion the cost may vary in between $5,000 to $85,000. It has the best results in low to medium contamination thickness. The energy consumption is around 0.18 Kw/h. The cost per piece is around $0.06 to $0.15 usd.

For medium and heavy duty immersion the cost may vary in between $5,000 to $85,000. It has the best results in low to medium contamination thickness. The energy consumption is around 0.18 Kw/h. The cost per piece is around $0.06 to $0.15 usd. (Knoth, Jackson, Sorbo, 2006).

Ultrasonic Cleaning

This method consist on tank filled with aqueous media that may or may not contains chemicals. The tank is connected to a motor that generates frequencies. These frequencies create bubble inside the tanks that impacts the surface of the piece removing the contaminant. The bubbles created depends on the frequencies. High frequencies create small bubbles and low frequencies bigger bubbles. The type and quantity of contaminants dictates the level of frequency used to clean piece.

The ultrasonic cleans contaminants such as paint, oil, grease, carbon, rust and oxidation. The cost of this method may vary from $10,000 usd to $180,000 depending on the accuracy, size and power. The energy consumption is about 0.86 Kw/h. per cycle but it varies depending on the frequency and time used. This type of method can clean regular and irregular shapes. Products with many holes or hidden chambers may be difficult to clean. The cost per piece varies around $0.035 to $0.45 usd. (Awad and Nagarajan) (Junzhong and Mingtao)

Molten Salt

Using the same principle of immersion cleaning, molten salt uses a bath of salts combinations and different temperatures to clean the surface. There three types of molten salt bath: 1) molten alkali metal nitrates or a mixture of nitrate ions. The temperature range from 150 °C to 550 °C. Have uses carbon, carbon alloy in addition to steel, vulcanization of rubber and in the burning off polymer residues; 2) Second type: Molten cyanide baths. The temperature range from 800 °C to 950 °C. This helps to hardness the surface subject to wear; 3) Third type: molten chloride salts. The temperature range from 700 °C to 1100 °C. It is used for tempering processes and neutral rinse baths to remove adhering cyanide or nitrate salts while maintaining the work at high temperatures. This method can be used for regular and irregular shapes and it is good cleaning pieces with small holes.

This type of cleaning is good to clean organic soils that forms in cars, trucks and plane’s engines. The cost has a range from $9,000 to $65,000 and energy consumption 0.1kw/h to 0.2 kw/h per cycle approximately. The cost per piece is around $0.6 to $0.8 usd. (Shoemaker, 2003)

Laser Cleaning

This method uses a laser beam to remove contaminant from the surface. The laser is directed by a mirror that gives the necessary direction to the beam. This method has a special feature. The energy of the laser can be changed to lower or higher levels. With this advantage, the contamination can be remove layer by layer in a controlled basis. Therefore, the system can be designed for a specific area and depth desired. The laser can be applied in a dry surface or liquid film at the surface to create a steam cleaning.

The best advantage of the laser cleaning is that has the power to remove any type of contaminant at any layer. The disadvantage is that irregular shapes cannot be clean, especially pieces with holes. The cost varies from $65,000 to $435,000 usd. The cost per piece ranges in between $2 to $3 usd. It has a high energy consumption from 2kw/h to 12 kw/h per cycle. (Tam, Park and Costas).

Vibratory Cleaning

Consist in a container filled with media that uses a device to apply time variable forces to the container to develop a periodic motion. The size and material of the particles the media can be changed to provide different results. If the contamination is thick, then a bigger and harder particle is used. If the contamination is thin, fine and soft particles may be used. Moreover, the type of material in the piece to clean dictates the type of media to use to avoid damages.

This method can clean pieces with regular shapes. It is not suitable for irregular shapes if it has small holes in it. It is considered more as a finishing process because cannot clean high levels of contamination. The cost may vary from $3,000 to $20,000 usd. The cost per piece is around $0.04 to $0.07 usd. The energy consumption 0.1 kw/h to 0.2 kw/h per cycle. (Ciampin, 2008)

Abrasive Cleaning

Abrasive cleaning consist in shoot particles into the piece to remove the contaminants in the surface. The process can be dry or wet. The main difference is that the dry does not use liquid or chemicals, just the dry particles. Dry abrasive uses sand, slags (copper, nickel, iron), minerals, glass, ceramic, sponges, pellets, natural products, carbon oxide or aluminum oxide grit. Wet blasting uses water with chemicals to remove the contaminants. It has different pH levels. Mostly uses: sand, mild alkaline cleaners, detergents, diluted acids, baking soda granules and other more. The abrasive cleaning may adapted to different needs by changing the nozzles on which the particles are shot. Bigger nozzles has more shotgun effect while a smaller nozzle is used for more detailed cleaning. It is important to be careful on the type of cleaner, pressure and application time because it may harm the piece.

This process has is good for regular and irregular shapes that does not have hidden chambers. Small holes may be hard to clean. The cost varies from $15,000 to $95,000, depending on the size and if it has the capability for wet and dry, only dry or only wet. The cost per piece ranges from $0.2 to $0.45 usd but it can go around $10 usd depending if expensive chemical are used, or if it is a big piece and needs more time to process. Abrasive cleaning is good for organic and inorganic contaminants. Examples includes: dirt, soluble salt, carbon, oxidation, paint, gaskets, rust, ash, or even a layer of the part’s surface. (Frenzel, 1999)

Thermal Cleaning

This method used high temperatures to burn the contaminants and convert it in ashes or gases. Normally, the method is performed in ovens. Convection oven is used when not direct flames cannot be used. The bottom part of the oven is heated and the radiated heat is what makes contact with the piece. Another one is the open flames oven. This type can reach higher temperature because of the exposure of the flames but can harm the piece. The main problems with this method are that due to high temperatures, the piece can melt, the material property may be changed or the piece can be damaged and not usable in the next process.

The process is good in removing organic contaminants, gasket material, rubber seals and heavy grease. The cost ranges from $8,000 to $55,000 usd. The cost per piece is around $0.08 to $1.55 usd. The consumption is around 0.3 kWh to 0.6 kWh. (Tanaka and Matsuoka)

Minor Cleaning Processes

This kind of process are used to perform the initial cleaning or to finalize cleaning from the surface of the piece such as: remove last part of the contaminants, clean residues, clean chemicals used on past cleaning method and/or perform the final details of the cleaning. This process includes spray wash, rinse, brush cleaning and any other that prepares the piece either to initial cleaning or finish it for the next step in the remanufacturing process.

Surface Contamination

The contamination it is formed through the usage of the piece. It is considered as a contaminant to any particle that may decrease the performance of the product or any particles that should not be in the surface. For example, the piston of the engine needs oil and grease to avoid high temperatures and wear due to the friction. But as time passes using the engine, oil and grease starts to stick in the walls of the piston chamber that mixed with the combustion and high temperature creates soot. This contaminant is not good to have it. Another example is painting and coatings. Even though that for the original product it is used to protect, when it comes to remanufacturing, those protection are considered contamination because a new paint and coat are needed to be applied. In order to have a “like-new” product, all the contaminants needs to be remove to have the real core material. Then, new paints, coatings and protections are applied.

According to the Nasa Handbook “Contaminantion Control”, there are five ways on which contaminants are created/deposited. 1) Gravity, the particle are deposited due the gravitational forces; 2) Electrostatic Charges, creates attraction of the particles in the process and the one surrounding that starts to form contaminations in the surface; 3) Physical entrapment, this type considers when a particle is trapped due to surface roughness, porosity, etc.; 4) Molecular Attraction, most common as a adsorptions and adhesions that exist in the contaminant’s particles that creates a coating/unwanted surface contamination; 5) Viscous Surface, due to grease, oils, etc., that mixed each other, either in some machining process or due to usage of the product.

A contaminant categorization is used to separate the different type of contaminants into different groups. The main reason to do it is due to the cleaning methods capacity to clean some type of contaminants better than others. The contaminants are categorized as organic and inorganic. Organic contaminant includes: organic particles, paints, lubricants, oils, grease, coatings, bacteria and fungi. Inorganic contaminants includes: oxide scale, wear debris, dust, moisture and inorganic lubricants (Long, 2013)

Linear and Non-Linear Programming

It is a mathematical technique to maximize or minimize functions, often called Objective Function, given some constraints and variables. The idea of the technique is to find the solution that meet the constraints and provides an answer to the objective function. Optimization it is post process to find the best possible solution for the objective function that obviously meets the constraints.

Programming it is called Non-Linear because the variable is modeled as power (square, cubic, etc.) or exist multiplication of variables, either in the objective function or the constraints (Bazaraa and Sherali, 2006)

A programming model consist in the following elements:

1) Objective Function: It is the function that is trying to maximize or minimize. In other words, it is the goal that is being trying to reach.
2) Decision Variables. These are the variables that plays the decision role. For example, on transportation problem, a decision variable may be the quantity of product to be sent on each route.
3) Constraints. This refers to the limit resources that exist in real life or lower and upper levels that is needed to maintain. Moreover, it exist the contingent constraints. These types of constraints are for special case, mostly use in binary programming, which makes the constraint "available" if the variable is chosen. To illustrate, let's take in consideration three projects. If a person chooses project one, then project three has to be selected. If choose project two, any other project does not have to be selected. So, the constraint plays the role if project one is selected. While not choosing project one, the constraint does not triggers.
4) Decision variable space. Notates the type of values that can be given to the variables and limit/boundaries on which the variables may have a value. For example, only Real Numbers that are positives ($Xij > 0 \ \forall \ i,j$).
5) Indexes. Refers to the different types of choices that exist in the problem. For example, in the cleaning process exist many type of cleaning method and there are many level of contaminants. In this case, the index "i" refers to the cleaning method and "j" to the level of contaminant, so the notation "C_{ij}" may refer to the cost of cleaning method 'i' for contamination level 'j'.

6) Feasible Region. It is the graphical region on which any point inside of it gives a feasible solution. The solution is optimal in a point or points of the region. Anything outside does not meet the constraints becoming a not feasible solution.

Fuzzy Set Theory (FST)

Fuzzy set theory allows imprecise and vague cases to become elements which can be precisely studied and analyzed. This type of set theory goes a step forward of the classic theory. Contrary to the classical theory (Boolean Value, Yer or No, True or False), FST accepts partial degree of membership. In classical theory a variable is either 1 or 0. This means that a variable that is part of a Set, it cannot be part of two subsets at the same time. For example the Set “Temperature”. This measures if something is hot or cold. These two levels are two different subset. If the room’s temperature is measured and gives a value of 50 °F, then the room is cold. The problem arises that 50°F it can be partially hot for some type of conditions. The classic theory does not deal with those kind of dilemmas. But in fuzzy sets, 50°F may be measure as 80% cold and 20% hot.

The importance of using this logic is that even though there is not exact values, a very accurate result can be achieved. Fuzzy sets was proposed for first time by Zadeh but fuzzy logic has was first developed by Lukasiewicz in the 1930s. The variables used in fuzzy set may have a value between 0 and 1, not only 0 or 1. The zero denotes that is no membership at all and 1 denotes full membership. Any value in between is considered a partial membership.

The elements that exist in a Fuzzy Set approach are: Set and Subsets, membership function, Fuzzy Inference (Fuzzy Rules) and Output. Sets and subsets are the available space on which a variable is being study. Membership function is to measure the level of membership that a variable has on a given subset. A subset are the different levels that are part of a set. A set it is the compound of subsets on which a variable may be part of it (Chen and Pham, 2000). For example, the set "C" can be denoted for cost. This set have three different subsets: Low, Medium, and High. Each level has a membership function/shaped defined. Fuzzy Inferences, or Fuzzy rules, are the different conditions that will happened if a variable takes values inside the any given subset. Most common are the If and Then rules. Output it is the final answer after the rules are applied (Chen and Pham, 2000).

The usage of fuzzy sets has four main steps: 1) Fuzzyfication of the variable (membership function), 2) Evaluation of the rules, 3) Fuzzy Inference; 4) Defuzzyfication output (Convert to a crisp value again with any method available, centroid is the most popular).

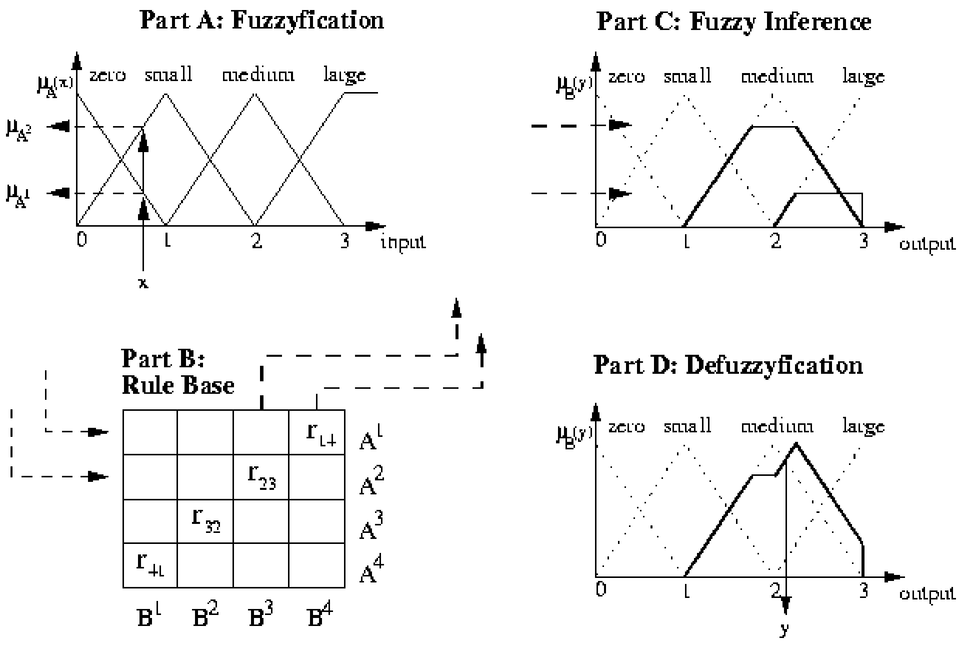


Figure 1 – Fuzzy Set and Fuzzy Logic Example (Brown and Harris, 1994)

Figure 1 denotes the process on which a Fuzzy set is performed. First the fuzzyfication of the crisp value. Then the rule base "If – Then" is evaluated. After evaluation, the fuzzy inference is performed to know the output. After the output is studied, the defuzzyfication takes place.

The application of this theory has grown dramatically. Nowadays, it has usage in control systems, decision making, artificial intelligence, operation research, linear programming, etc. One important application is the use of fuzzy logic in Supply Chain Management. This method can be used to evaluate suppliers by creating a decision support system. Kumar and Sing, 2013 developed decision model based on 88 Indians textile industries to evaluate suppliers. Also, fuzzy sets in supply chain helps to optimized supply chain solution when dealing with uncertainties (Peidro, 2009). These includes supply, process and demand uncertainties (Davis, 1993).

Genetic Algorithm (GA)

GA it is a technique in the field of artificial intelligence that helps to solve problems on which normal approaches are to slow or failed to provide and exact answer. GA may be defined as a "*search technique used in computing to find true or approximate solutions to optimization and search problems."* (Harrim, 2013). The GA was invented for first time by John Holland in the early 1970's. GA has gained high reputation and it is used in many areas because helps to solve complex and non-linear optimization. This method simulates the Darwinian evolution (survival of the fittest). By this simulation, the array of solutions that are closed to the optimal one may be considered as the fittest (or survivors), and those far from the optimal characteristic are disposed.

On important point is to be careful about generation of the algorithm. It is important to have constraints to assure that the solutions provided are feasible for the problem.

The procedure of GA consist in the following (Tiwari and Harding, 2011)

1) Fitness Calculation: consist in the calculation of the objective value with the solutions gathered in the solution space. When a solution is better than the best one found previously, the fitness value (objective) is improved.

2) Penalty Mechanism. Consist in giving a penalty (e.g. extra cost when maximizing profit) in order to decrease the probability of this type of solution in the next generations. Also, helps to penalize when a constraint it is not met.
3) Repair mechanism. In order to avoid infeasible solutions, it is necessary either a constraint or a mechanism that evaluates value of the variable that in case it violates the constraint, change to another value that is inside the feasible boundary.
4) Selection Procedure. There many method in this case. The selection procedure consist in choose the next set of solutions from what is called the "parent solution" (that is no more than a first set of solution).
    a. Roulette wheel. It is method of selection on which the next set of solution is set by the probability factor of each solution.
    b. Crossover. Consist in an exchange of solution between two "parents solution" to create "child" solutions. This consist in a point of the string on which the next alleles are interchanged. It is necessary to set a probability of crossover on which that number of solutions will go to a crossover procedure.
    c. Mutation. Consist in randomly change a value inside the string to *"create new possibility of exploration of solution space"* (Tiwari and Harding, 2011).

Other important GA's procedure in creating solutions are Elitism and Rank Selection (Mohebbi, Taheri, Nooshiravani, 2011). Elitism consist in retain a pool of individuals that gave the best solution in each generation. This is to lower the risk of losing good solutions when the strings are set to crossovers and mutation operations. This method was provided by Kenneth de Jon in 1975. Rank selection in the other hand, the solutions are ranked according to the fitness expected. Now, the strings are not evaluated not for their quality of the fitness but for the rank that they have. This avoids too quickly convergence to one type of solution. This method was proposed by Baker in 1985.

Fuzzy Sets and Genetic Algorithm have been used lately to create solution to complex problems with high grade of uncertainty. Gerardo Acosta and Elıas Todorovich (2003) created a mix of fuzzy control and GA for industrial applications on which the components of the fuzzy sets were optimized with the application of GA. Phillip, Karr and Walker developed fuzzy controller capable to maneuver a helicopter. The GA was used to discover fuzzy rules that best applies to their need.

Fuzzy control with genetic algorithm it is a very important matter in which a lot of investigation have been performed. Most of the studies have been made for exactly measures using sensors and metrics to evaluate the rules. With GA, the optimization of those control values creates better fuzzy rules.

Moving to another area, Rajangam and Arunachalam proposed Fuzzy Logic Controlled Algoritm to EDL. They used this method to optimize the combination of power outputs for all generating units that leads to a minimization of the total fuel. This approach it is more linked to purpose of this study on which decision is made using FS and GA to optimize the resources available.

As it can be seen, FS and GA are great combination to develop controls and decisions making. This approach may be used to better understand the uncertainties that exist and to optimize the current solutions inside remanufacturing process.

CHAPTER III

DECISION SUPPORT MODEL FOR PROCESS PLANNING

This chapter contains the explanation step by step of the model proposed for the decision making tool for process planning in remanufacturing. The section consist in four parts. First, the mathematic programming model in which decision variables, objective function and constraints are explained. Second, the fuzzy sets with the membership functions. Third, the genetic algorithm is explained and how the search algorithm works for the model proposed; and fourth, the decision support model is explained in a flowchart. It is important to note that these Fuzzy sets and Membership function were implemented to the best understanding of the current general knowledge. Every remanufacture process has different criteria, different measurements and different needs. The decision support model proposed should be used as a general model to help the remanufacturer make a more accurate decision, however, the technical information should be modified according to their own process.

The following terms are used throughout the model.

- Sequence of cleaning processes (Y_i). Refers to the cleaning chosen to perform the study. The index "i" is used to differentiate from other sequence. It is modeled as an array of processes that contains the sequence. $Y_i = [y_1 \ldots y_j]$. For example, $Y_1 =$ [1 2 3 4], in which the sequence is Ultrasonic, Abrasive, Laser and Thermal. Each jth term represents one cleaning method and the ith array represents the sequence in which will be performed. This variable is created randomly.

- Process Time (T_i). Similar to Y_i, it refers to the time in which each cleaning process will be used. It is also an array and each position refers to the time that a process of the array Y_i will be used. $T_i = [t_1 \ldots t_j]$. For example, $T_1 = [15\ 20\ 25\ 30]$ refers to process time of the cleaning processes. Also, the index T_i has a relationship to the index Y_i. Each j^{th} term represents the time of each method and the i^{th} array represents the process time.
- Acceptable Process Time (Tw). This is the maximum process time allowed for the combination of cleaning process. It is given by the user by a number or by a subset of the set Time. It serves as an upper limit.
- Acceptable Clean Level (CLp). It is minimum cleaning level that the piece should meet in order to be suitable for the next processes. It is given by the user by a number or by a subset of the set Cleanliness Level. It serves as the lower limit.
- Acceptable Energy Consumption (Ev). Similar to time Tw, Ev it is the maximum energy consumption that can be used in the cleaning process. It is given by the user.
- Fuzzy Inference Output (P_i). Refers to the results given the fuzzy inference after evaluating the necessary rules. The output is Cleaning Result (CL), Energy Consumption Result (EC) and Cost Result (CR). Later in the process, the Y_i and T_i are combined to check the constraints with the input given by the user.
- $\overline{y_J}$ represents the binary element for the array Y_i. If an element of the array is bigger than zero, the $\overline{y_J} = 1$. If not, $\overline{y_J} = 0$
- Contamination Type *(CT)*. It is a user input that refers to the type of contaminant that is located in the surface of the piece. May be organic, inorganic, mixed, etc.

- Contamination Thickness *(CT)*. It is a user input that refers to the amount of contaminant located in the surface of the piece.
- Material Type *(MT)*. It is a user input that describes the type of material that is contained in the piece. The material may be metal, ceramic, polymer, etc. For the study, the material type may be metal or non-metal.
- Piece Shape *(PS)*. It is a user input that gives information about the form of the piece. It may be flat, round, may contain holes. For the study, *PS* is used as a fuzzy set with sub-sets of Simple, Complex and Very Complex.
- Feasible Solutions *(Q_i)*. This term is an output after checking the constraints. It contains information about the inputs of the user, sequence of cleaning processes, processing time and output of the fuzzy inference. This array is built each generation with the processes that only met the constraints.
- Best solution *(s_i)*. Refers to the best solution of each generation that is contained in Q_i

The Model

Decision Variables

The decision variables are Y_i and T_i. These two variables gives the sequence of process and the time in which process of the sequence is performed. They are the responsible, with the user input, to evaluate the performance of the process in terms of cleanliness level, energy consumption and cost. After the evaluation is performed, it is important to calculate the cleanliness level reached, the energy consumed and the cost incurred. The first two and the T_i are used in the constraints checking to assure that the sequence of processes are suitable.

Constraints

The constraint in the model serves as filter to disregard any options that is not feasible, in other words, that does not meet the user expectation (given in the input). *CLp, Tw* and *Ev* are limits that needs to be met. *CLp* is the minimum cleaning level, or lower limit that should be met. *Tw* and *Ev* represents the maximum limit (or total resources available) for time and energy consumption of the process. The user implements the constraints when he gives the input of the Acceptable Process Time, Acceptable Cleaning level and Acceptable Energy Consumption. The output P_i = [CL EC CR] given by the fuzzy inference is used to check the constraints. CR (Cost Result) it is not part of the constraints, but it is part of the array. The time constraint it is checked with the array $T_{i.}$

As explained before, CL it is the cleaning level reached by the sequence of process. The cleaning output has to be higher than the one given by the user. The output EC and the process time generated T_i are needed to be lower the user input *Tw* and *Ev*. The constraints are set as follows:

$$CL \geq CLp \tag{1.1}$$

$$Ti \leq Tw \ \rightarrow \sum_{j=0}^{j=n} \overline{y}_j t_j = R_i \ \leq Tw \tag{1.2}$$

$$EC \leq Ev \rightarrow \sum_{j=0}^{j=n} \overline{y}_j E_j \ \leq Ev \tag{1.3}$$

Objective Function

The objective function is minimize the cost given by the function:

$$Min\ (Z) = \sum_{j=0}^{j=n} \overline{y}_j C_j \ \rightarrow Min\ (Z) = CR \tag{1.4}$$

Fuzzy Sets and Membership Functions

The fuzzy sets and the membership function are used to explain first the space on which a variable is contained and to convert a crisp value to a specific degree of membership. It is called fuzzy sets because the "space" that is contained in the set cannot have specific value, instead, degree of membership. Subset are all the different categorization that may exist and be contained in the set. Membership function is an indicator of measure of how much a variable is contained in a specific subset. If a variable has a membership of 1, then the variable it is contained completely in the subset. If the variable has a membership of 0, then it is not contained in the subset. Now, if the variable has a membership between 0 and 1, then the variable is partially contained in the subset.

The sets used for the decision model are:

- *Contamination type (CT)*. This set is used to determine the category of the contaminant. It is divided into: Organic, Inorganic and Mixed. For the model, these are mutually exclusive. The contamination is either organic, inorganic or mixed. The input is a discrete value where 1 = Organic, 2 = Inorganic, 3 = Mixed. Organic contamination includes: organic oils, grease, paints, dirt, etc. Inorganic refers to: dust, rust, resins, etc. Mixed it is a mixed between organic and inorganic. (L. Mihan) (Chuan Sheng Si)
- *Contamination Thickness (CTh)*. The set describes the amount of contaminant that is in the surface of the piece that. For the model, the set has three subsets: Low contamination, Medium contamination and High contamination. The range used is between 0 and 4. Where 0 is the lowest level of contamination and 4 the highest level. The input is discrete number in which: 0 = Very Low contamination layers, 1 = Low contamination layers, 2 = Medium Contamination layers, 3 = High contamination layers and 4 = Very high

contamination layers. The membership function is the following. It is used a triangular membership function for the simplicity. The levels of contamination thickness are set to those sets by research of the literature. The sets Low, Medium, High and Very high are used to explain the level of contamination in the piece

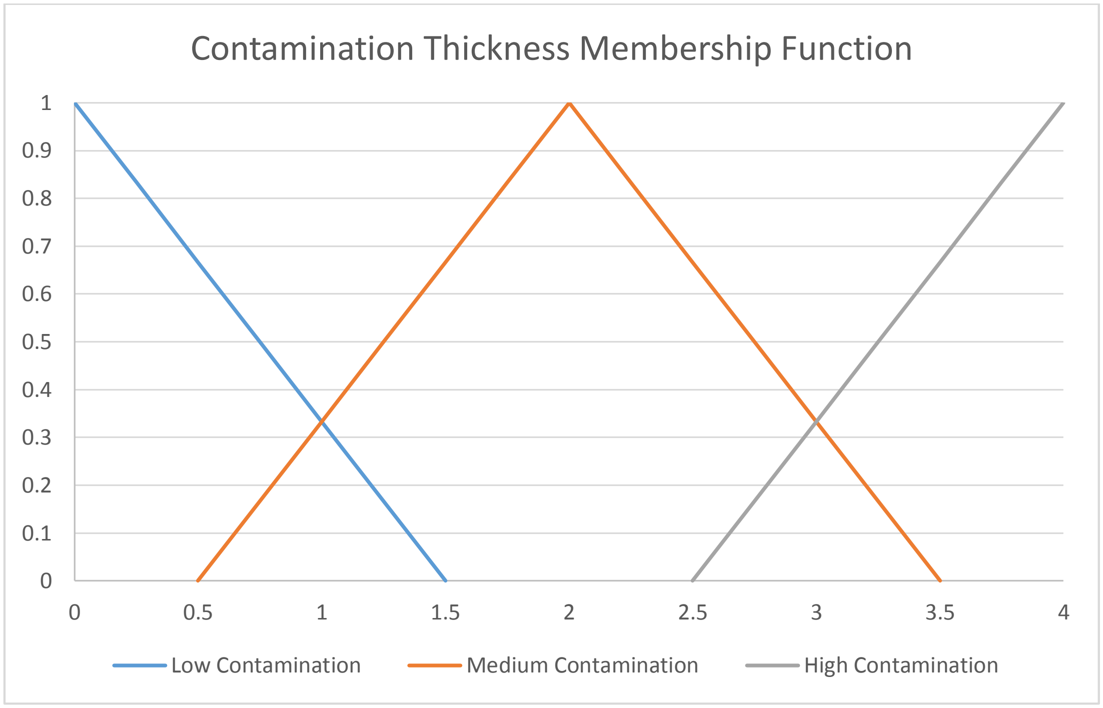


Figure 2 – Contamination Thickness Membership function

- *Product Shape (PS) and Material Type (MT)*. This relates to properties of the product. It is important to understand the shape/size of the product and also the material composition on which was built. The main reason is because different cleaning methods supports different kind of materials and shapes. Some method cannot support complicated shapes or with holes because it does not have the capability to mold according the piece. Another methods use high temperature that may change the material properties and also, there is a risk of melting the product. The material type is Boolean value were 1 = Metal and 2 = Non – metal. After carefully reading the literature of the different cleaning processes, the sets used to explain the shape, the proposed set for Piece Shape consists in three subsets: Simple, Complex and Very Complex. The triangular membership function is used to simplify the process.

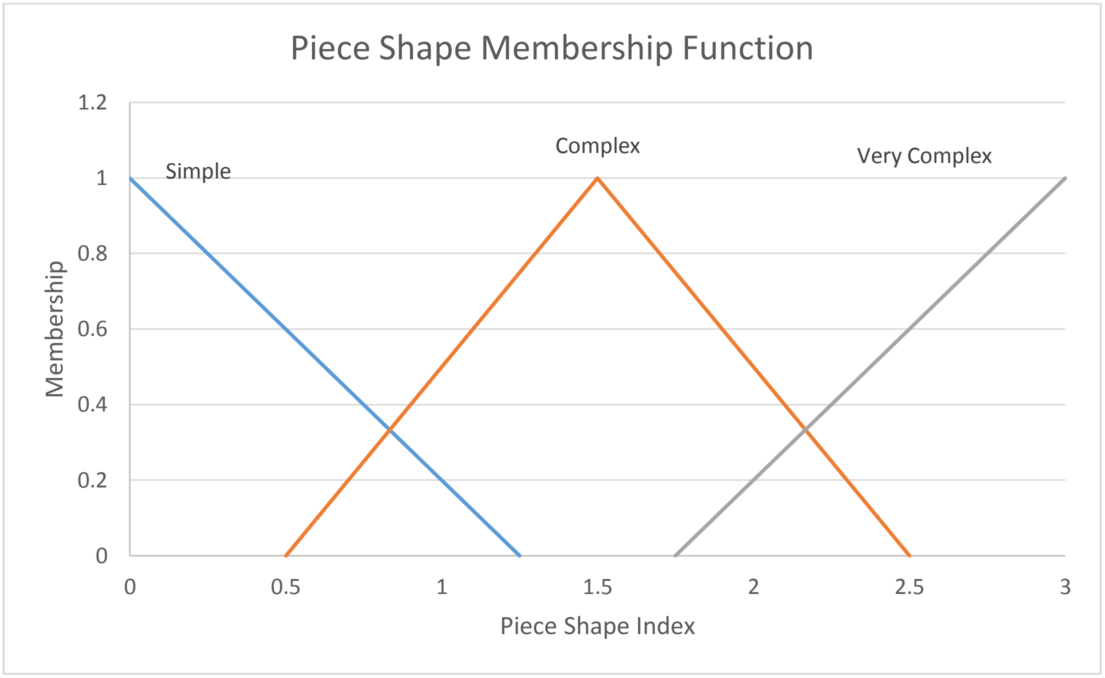


Figure 3 – Piece Shape Membership Function

- *Time*. This set is used for the process time and acceptable process time defined by the user. Also, the time is used to dictate the time that the process is running. Both uses the same membership. The different between the user input and the processing time by the cleaning method is that the time given by the user is for constraints purposes (Total processing time) and the time used for the cleaning method is for “cleaning processing” time. The set Time has the following membership function. Because the different processes has different processing time in which are effective, it is decided to use a range between 0 and 60 minutes. This range comes from the literature of the different cleaning processes on which is specified the time range in which the cleaning is effective. The triangular shape for the membership function is used to simplify the process.

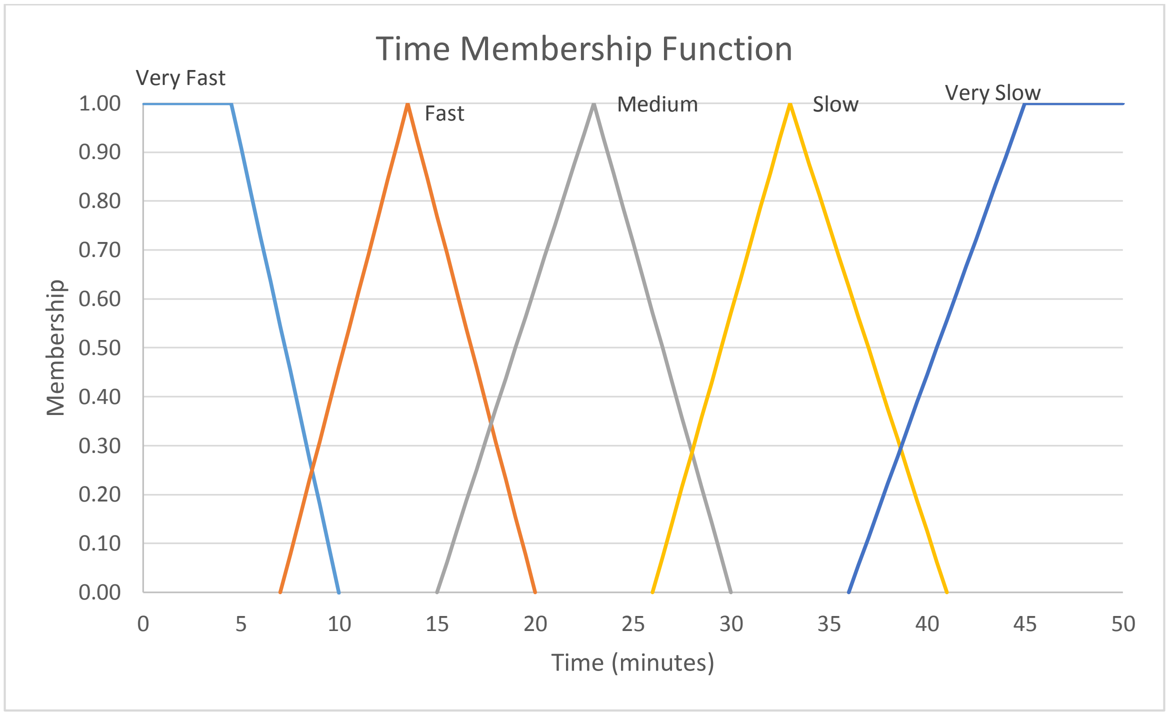


Figure 4 – Time Membership Function

- *Cleanliness Level*. This set refers to the overall cleaning output after a cleaning method was used. Similar to the contamination thickness, the cleanliness level is check in the surface of the piece and measure how clean it is the piece. The assumption is that with less contaminants or undesired components in the surface, the piece is cleaner. The range proposed for the model is between 0 and 10, where 0 it is not cleaned at all and 10 is completely clean. The numbers in between refers to partial cleaning in which some contaminants were removed but there are still unwanted components in the surface that needs to be cleaned. This set is used to evaluate the user input to the model and then used as constraint in order to evaluate the cleaning performance of the different cleaning methods. The set Cleanliness level has a triangular membership function to simplify the process. The range between 0 and 10 it is proposed in this research to rank the level of cleanliness. It is a numerical index measure to grade the level of cleanliness achieved by the process and to set the cleaning level constraint.

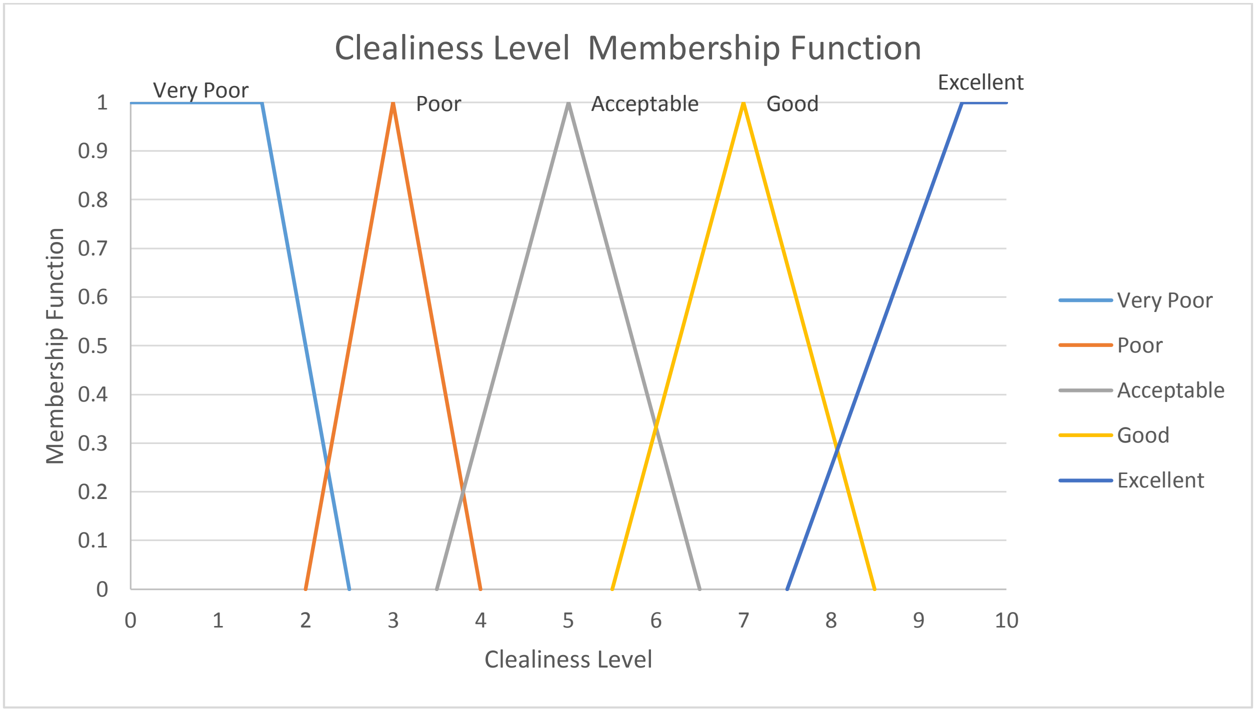


Figure 5 – Cleanliness Level Membership Function

- *Energy Consumption*. This set refers to the energy consumed by the different cleaning methods when are working. Also, it refers to the user input for the expected energy consumption that wants to be consumed. The user input is used to evaluate the energy consumption constraint (explained later in the programming model) with the energy consumed by the cleaning processes. The energy is measured in kWh. It has the following membership function.

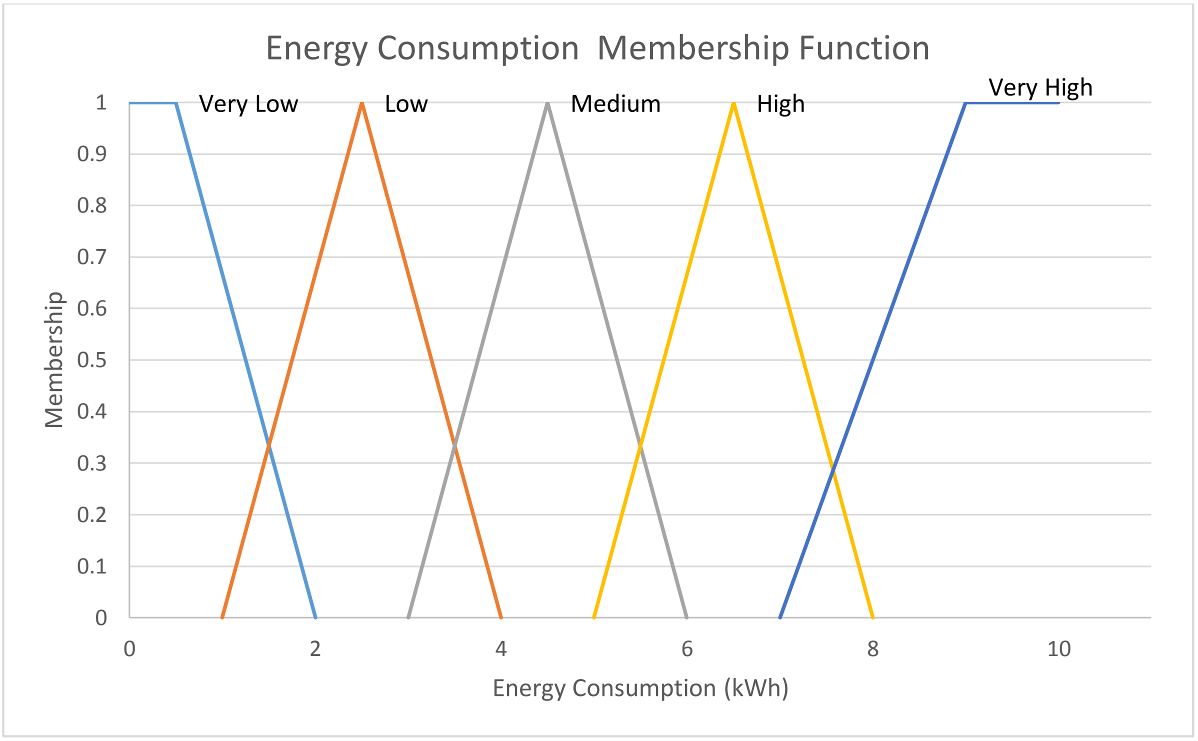


Figure 6 – Energy Consumption Membership Function

- *Cost*. This set refers to the cost per cleaning process and the overall cost. This become the objective function of the model. The set is divided into five subsets that are: Very Low, Low, Medium, High and Very High. It ranges from $10,000 to $460,000. The cost includes designs, machines, implementation of the process and every cost that is incurs when implementing the cleaning process. It can be refer as “system cost” also. It does not include the cost per piece. The set Cost has a triangular membership function to simplify the process.

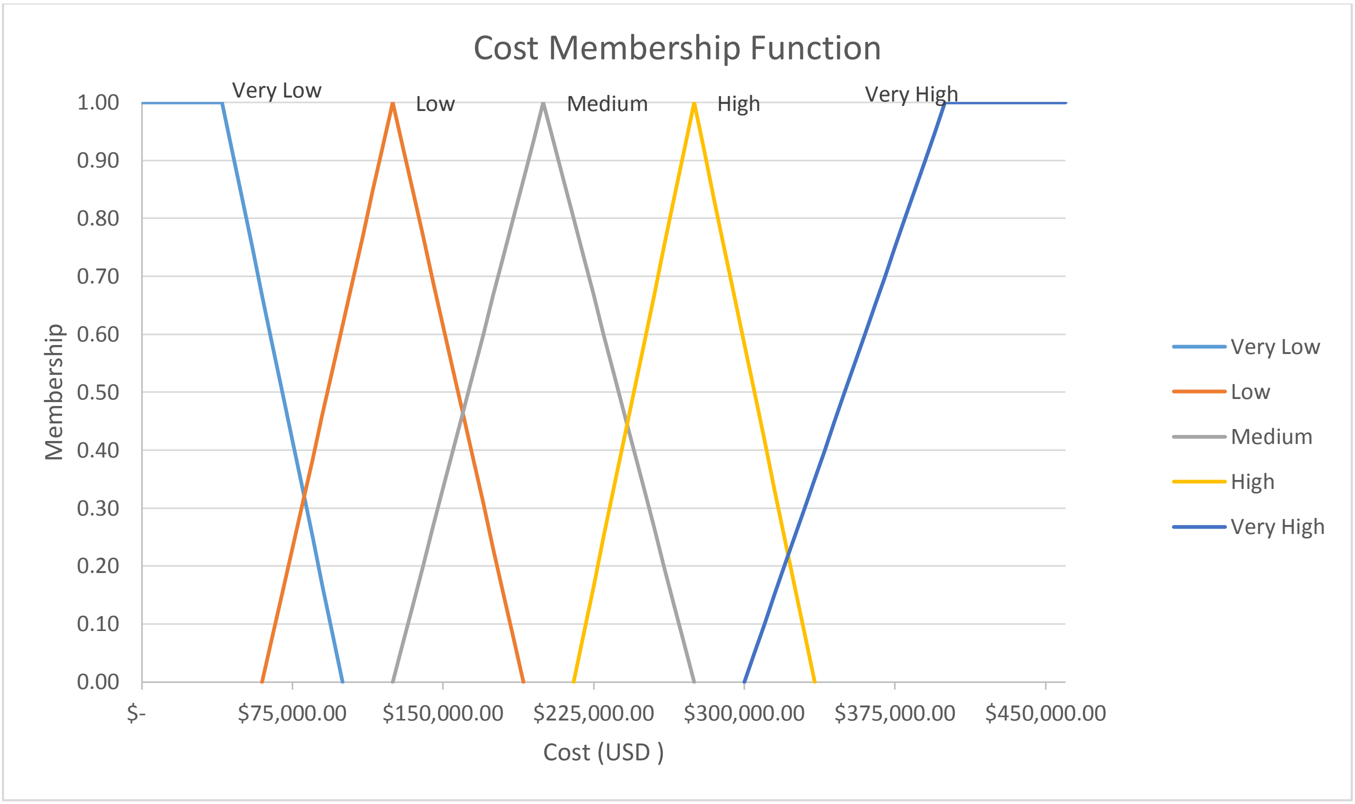


Figure 7 – Cost Membership Function

Fuzzy Inference Rules

In order to predict the performance of the cleaning process, it is necessary to set the rules that will dictate the performance. The fuzzy inference is used because there is no mathematical model that relates to cleaning performance, energy consumed and cost. There are ranges on which the cleaning process delivers a good result. The elements of the fuzzy inference are: input, implication, output.

The input for the inference rule is: Contamination type (CT), Contamination Thickness (CTh), Material Type (MT), Piece Shape (PS), Cleaning process (y_j) and Processing Time (t_j). CTh, PS and t_j, are fuzzified using the membership functions shown before and put into linguistic term. The outputs are Cost Result (CR), Cleanliness level (CL), Energy Consumption (EC).

The Fuzzy Inference has its literature background in different articles where the cleaning processes were studied and by the best understanding of the authors. Nonetheless, every user may modify, change, eliminate or add rules depending on their need. The implication consists in the relationship between the inputs and the outputs. For the model, the rules consist in the following:

"IF contamination type is Organic and Contamination Thickness is Low, and Material Type is Metal, and Piece Shape is Simple Cleaning Process is Ultrasonic, and Processing Time is Fast, THEN, Cost Result is Low, and Cleanliness Level is Excellent, and Energy Consumption is Low"

In case that a process it is not possible to clean a specific type of contaminant, contamination thickness, shape or material, the predicted performance it is going lower. For example. Ultrasonic cleaning deals with low to medium contamination. In case that the piece has high contamination, the fuzzy rule is:

"IF contamination type is Organic and Contamination Thickness is ***High****, and Material Type is Metal, and Piece Shape is Simple Cleaning Process is Ultrasonic, and Processing Time is Fast, THEN, Cost Result is Low, and Cleanliness Level is* ***Poor****, and Energy Consumption is Low"*

Because the Ultrasonic is used for the same time, the system cost and energy consumption will not change because the processing time is the same. The performance decreases in the cleaning result. All the different rules are built like these ones.

There are combinations that are not feasible. One cleaning process may not support on type of contamination. For those, the fuzzy rule is to make it infeasible. For example, thermal cleaning does not support inorganic contamination. The performance is set to the worst levels in each case to assure infeasibility in the constraint checking.

This type of rule is set to all the different possible implications. This rule relates to the initial condition of the piece, the type of cleaning process and the amount of time that is going to be put to cleaning. The result is in term of how much is going to cost use that cleaning process during the specified time, the energy consumed during that time and the quality of the cleaning process.

Looking in detail, Contamination Thickness, Piece Shape and Processing Time can be converted from crisp value to linguistic term using the membership functions. The outputs: Cost Result, Cleanliness Level and Energy Consumption are put in crisp value via defuzzifcation (centroid). In order to perform the defuzzification, it is necessary the membership function of each one of the outputs shown before.

Genetic Algorithm and Genetic Operators

The genetic algorithm serves as optimization tool. The GA generates the possible solutions contained in Y_i *and* T_i. These are going to be a random integer number between 0 and n, where "n" represents the number of potential cleaning processes that may be used, each one represented by y_j, For T_i, the elements has a random generation between 0 and "m", where m represents the maximum time range.

Crossover and mutation have a rate given by the term *Pc* and *Pm*. The first one is set around 0.7 and the second one around 0.1. In other words, 70% of the feasible solutions are going to be set for a crossover operations and 10% of those are going to mutate. The mutation is set to be randomly to any j^{th} of the arrays Y_i and T_i. The crossover is performed between the best "s" solutions to generate new population. The term "s" stands for a number of best solutions. It may be 10, 20, 30 or any number that represent the best solutions. This is set by later in order to optimize the computation time. The crossover point is set randomly. Before performing any operations, the best "s" solutions are going to be saved. The best solutions may be contained in past generations.

All new generations enter in the loop to evaluate the process performance, the constraints and save the solutions of the feasible ones.

The stopping rules used in the GA are going to be the following:

- The number of generations are met.
- The objective function has not improved in the past two generations or is between the +/- N%. The "N" stands for the tolerance of the user.

Inputs

The user input is an array named I = [CT CTh MT PS T_w CL_p E_v] which consist in contaminant type, contaminant thickness, material type, shape of the piece, expected process time, expected cleaning level and expected energy consumption. The first four elements of the array are used to perform the fuzzy inference engine. The last three are to evaluate the constraints. The user needs to input these information in order to calculate the best possible combination of cleaning processes that results in the lower possible cost.

Generation of Seeds

The seeds are generated randomly for Y_i and T_i. Each element of the array is random integer number between 0 and n, representing the different cleaning processes available to choose from. If the remanufacturer has 4 cleaning processes available, then n = 4. The zero should be included to represent that "no cleaning process" is used. Similarly, T_i has random numbers between 0 and m, where "m" represents the upper limit for the time range. Later, this number are turn to linguistic terms with the fuzzy inference rules.

Process Evaluation Loop

After gathering all the inputs and the seeds are generated, it is necessary to evaluate the performance of sequence of the cleaning processes dictated by $Y_{i.}$ The inputs to this operation are:

CT, CTh, MT, PS, y_j and t_j. The fuzzy inference engine does the performance evaluation giving the result in terms of cleaning level, energy consumption, cost and a term called "New contamination thickness" with the name $I_{nct,}$. This new term serve as an input for the next element in the $Y_{i.}$ After the first element of the array is evaluated, the results needs to be storage and evaluate the next element of Y_i. The input for the evaluation of the next element are: CT, $I_{nct,}$ MT, PS, y_{j+1} and t_{j+1}.

The increment "j+1" is to evaluate all the elements in the arrays Y_i and *Ti.* In the case than an element of the array is equal to zero, the solutions for that given iterations is going to be zero. After all the elements are evaluated, the final result for the array Y_i is storage in the array P_i that was explained before.

Constraints Checking

After gathering all the results from the process evaluation loop, these ones are checked to know if they satisfy the customer requirements. In other words, to know if the constraints are met.

As it was explained before, *CLp*, *Tw* and *Ev* are the user input and represent the limits of the user requirements. *CLp* is a lower boundary. *Tw* and *Ev* are the upper boundary. The constraints are checked by the equations 1.1, 1.2 and 1.3. Any array Y_i that does not meet any of the constraint is disregarded and no longer taken in consideration as a feasible solution. This step of the model is to filter which solutions that are suitable from those that are not. Finally, the objective function is calculated and it is stored as part of the array *"Q_i"*. At the end, this array has all the information from the user input to the overall results.

$$Q_i = [CT\ CTh\ MT\ PS\ Y_i\ T_i\ CR\ R_i\ CLp\ Ev]$$

Genetic Algorithm Process

This is the last part of the loop that closes the model and chooses the best solution. The Genetic Algorithm Process (GAP) has the stop conditions mentioned before. All the solutions that are crossover and mutated enter again in the model to evaluate the performance and check the constraints. When the stopping conditions are met, the best solution is chosen and it is given to the user. The model flowchart is shown below

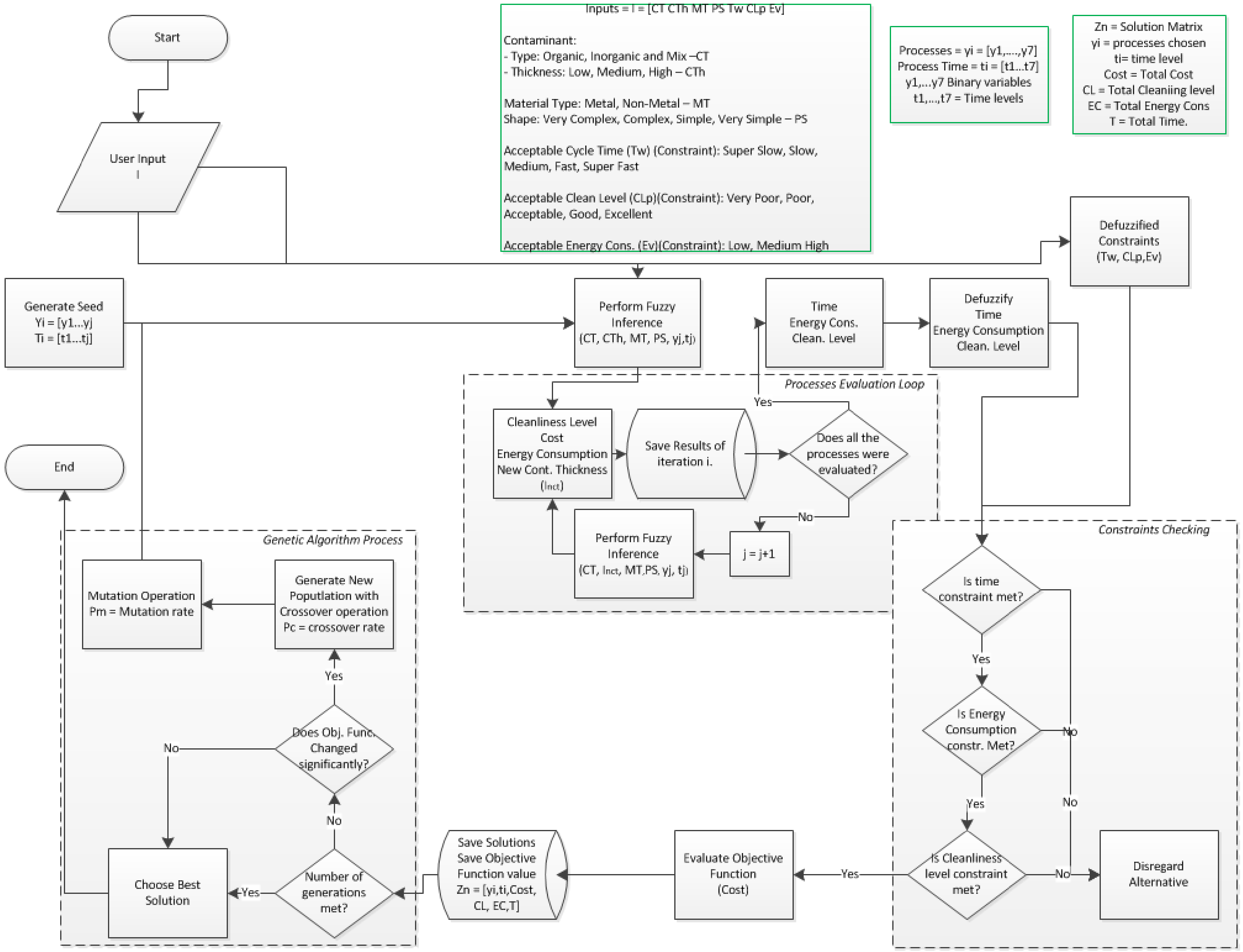

Start
User Input
I
Inputs = I = [CT CTh MT PS Tw CLp Ev]
Contaminant:
- Type: Organic, Inorganic and Mix –CT
- Thickness: Low, Medium, High – CTh
Material Type: Metal, Non-Metal – MT
Shape: Very Complex, Complex, Simple, Very Simple – PS
Acceptable Cycle Time (Tw) (Constraint): Super Slow, Slow, Medium, Fast, Super Fast
Acceptable Clean Level (CLp)(Constraint): Very Poor, Poor, Acceptable, Good, Excellent
Acceptable Energy Cons. (Ev)(Constraint): Low, Medium High
Processes = yi = [y1,....,y7]
Process Time = ti = [t1...t7]
y1,...y7 Binary variables
t1,...,t7 = Time levels
Zn = Solution Matrix
yi = processes chosen
ti= time level
Cost = Total Cost
CL = Total Cleaniing level
EC = Total Energy Cons
T = Total Time.
Defuzzified Constraints (Tw, CLp,Ev)
Generate Seed
Yi = [y1...yj
Ti = [t1...tj]
Perform Fuzzy Inference (CT, CTh, MT, PS, yj,tj)
Time Energy Cons. Clean. Level
Defuzzify Time Energy Consumption Clean. Level
Processes Evaluation Loop
Cleanliness Level Cost Energy Consumption New Cont. Thickness (Inct)
Save Results of iteration i.
Does all the processes were evaluated?
Yes
No
j = j+1
Perform Fuzzy Inference (CT, Inct, MT,PS, yj, tj)
End
Genetic Algorithm Process
Mutation Operation Pm = Mutation rate
Generate New Poputlation with Crossover operation Pc = crossover rate
Yes
Does Obj. Func. Changed significantly?
No
No
Choose Best Solution
Yes
Number of generations met?
Save Solutions Save Objective Function value Zn = [yi,ti,Cost, CL, EC,T]
Evaluate Objective Function (Cost)
Constraints Checking
Is time constraint met?
Yes
Is Energy Consumption constr. Met?
No
Yes
No
Is Cleanliness level constraint met?
Yes
No
Disregard Alternative

Figure 8 – Decision Model Flowchart

CHAPTER IV

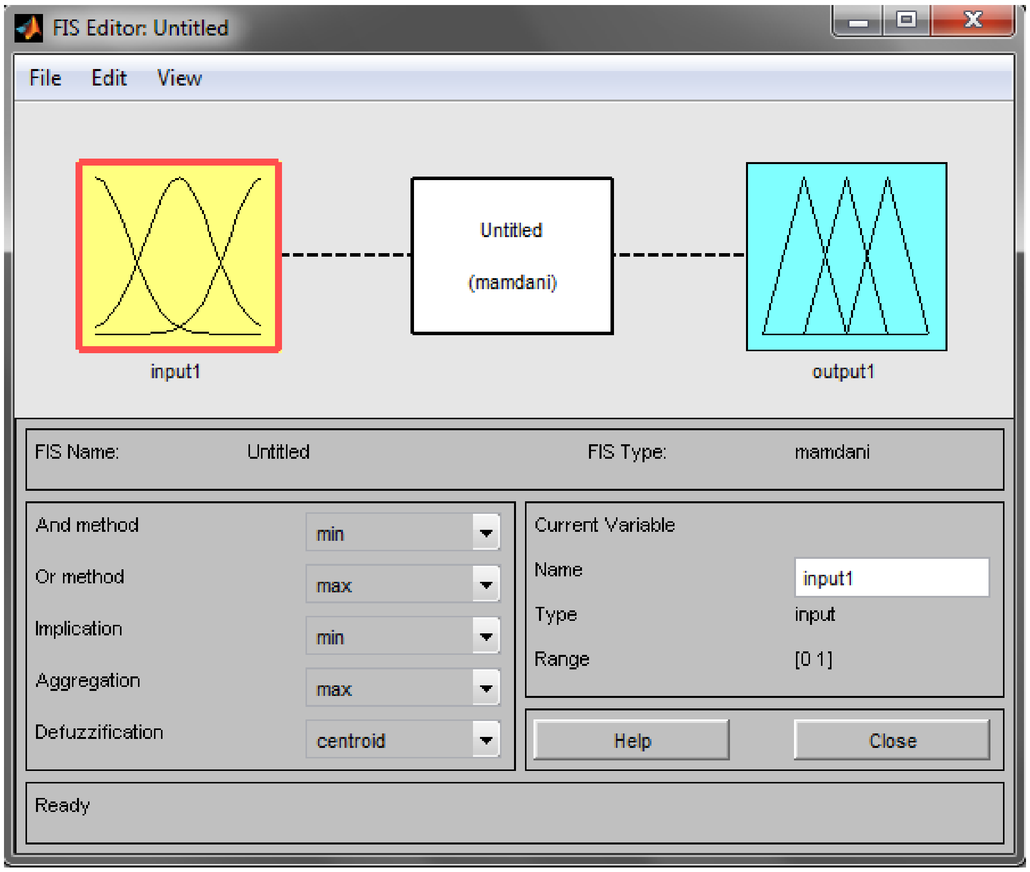
SOFTWARE, CASE STUDY AND DISCUSSION

This chapter discusses the software developed for the decision support in process planning for remanufacturing and demonstrates an example to show how the software works.

Software – Decision Making Tool for Cleaning Processes in Process Planning

The software is built in the Matlab® due to the simplicity of manipulating matrices, new variables and mostly, the "Fuzzy Toolbox". This toolbox is very simple to use and supports calling inside a Matlab® script.

Figure 9 shows a screenshot of the Matlab® fuzzy toolbox. It consist in three main parts: inputs, fuzzy inference, outputs. The inputs are all the variables given by the user that are considered to study. The fuzzy inference is where all the If – Then rules are established. Output are the final answer of the case or cases evaluated. Also, the toolbox leave the user to decide different parameters such as: defuzzification method, aggregation method, implication method, etc. For the software all the parameters are used in their default values. Also, the membership function of the inputs and outputs are very easy to set up.



FIS Name: Untitled
FIS Type: mamdani

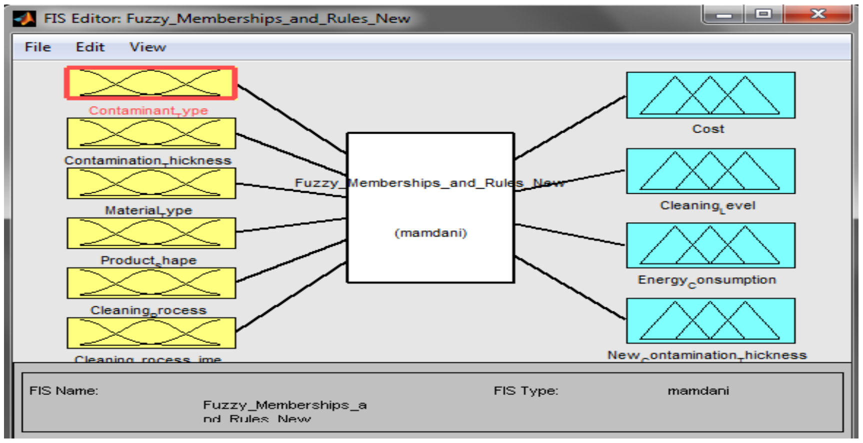
And method: min
Or method: max
Implication: min
Aggregation: max
Defuzzification: centroid

Current Variable
Name: input1
Type: input
Range: [0 1]

Ready

Figure 9 – Matlab® Fuzzy Toolbox

In Figure 10, it is shown the inputs and outputs used for the fuzzy toolbox evaluation. The inputs are: contamination type, contamination thickness, material type, piece shape, cleaning



FIS Name: Fuzzy_Memberships_and_Rules_New
FIS Type: mamdani

Figure 10 – Inputs and Outputs variables

processes and processing time. The last one refers to the amount of time on which a cleaning processes is used. The outputs of the model are cost, cleaning level, energy consumption and thickness of contamination remaining after the process. The last one is used to determine the new level of contamination on which the new cleaning process will start cleaning because only part of the contaminants was cleaned before.

Figure 11 shows and example of the membership functions and figure 12 shows the fuzzy inference process. The process is relatively simple. The user just clicks the different inputs that are going to be used in the rule and then clicks on the outputs that should be met when the rule is evaluated. For example, “IF contamination type is Organic, and Contamination thickness is Low, and Material type is Metal, and Piece Shape is Simple, and Cleaning Process is Ultrasonic and processing time is Fast, THEN, Cost is Low, Cleaning level is Excellent, Energy Consumption is Low and New Contamination Thickness is Super Low.

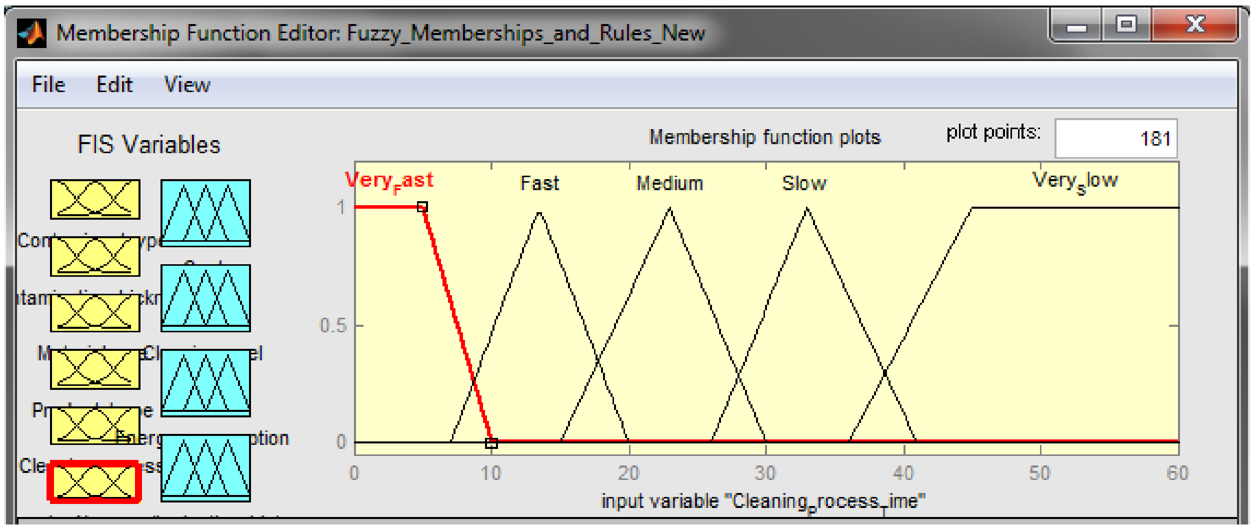
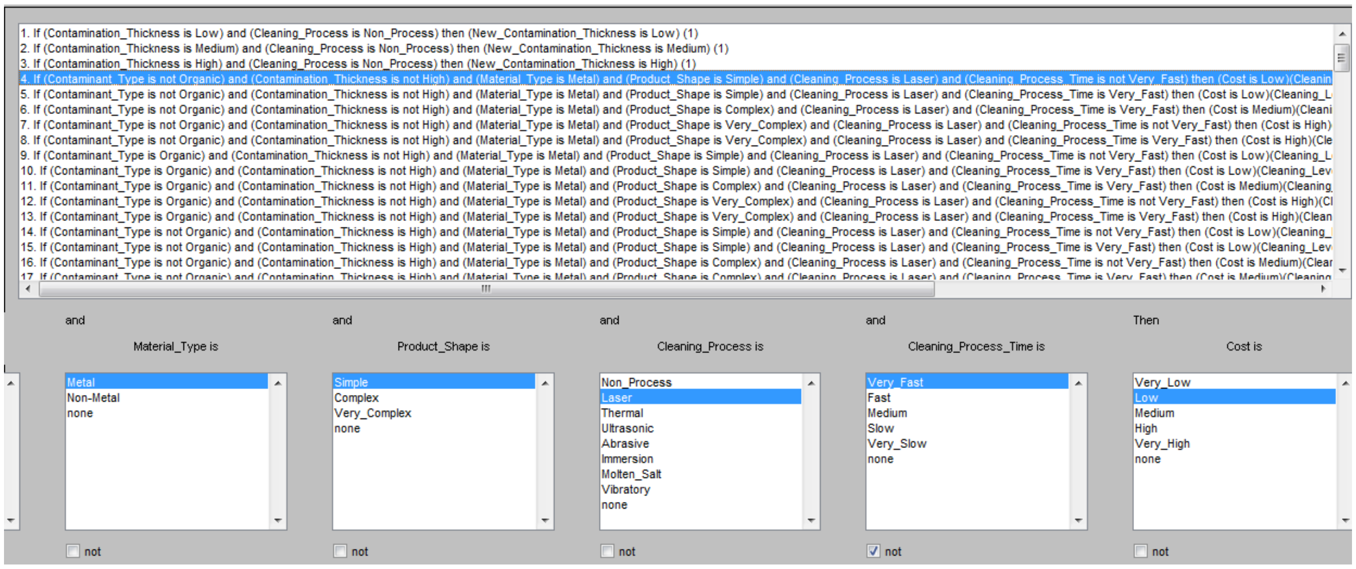

Membership Function Editor: Fuzzy_Memberships_and_Rules_New
File
Edit
View
FIS Variables
Membership function plots
plot points:
181
Very_Fast
Fast
Medium
Slow
Very_Slow
1
0.5
0
0
10
20
30
40
50
60
input variable "Cleaning_Process_Time"

Figure 11 – Example of the membership function in the Fuzzy Toolbox®



```
1. If (Contamination_Thickness is Low) and (Cleaning_Process is Non_Process) then (New_Contamination_Thickness is Low) (1)
2. If (Contamination_Thickness is Medium) and (Cleaning_Process is Non_Process) then (New_Contamination_Thickness is Medium) (1)
3. If (Contamination_Thickness is High) and (Cleaning_Process is Non_Process) then (New_Contamination_Thickness is High) (1)
4. If (Contaminant_Type is not Organic) and (Contamination_Thickness is not High) and (Material_Type is Metal) and (Product_Shape is Simple) and (Cleaning_Process is Laser) and (Cleaning_Process_Time is not Very_Fast) then (Cost is Low)(Cleanin
5. If (Contaminant_Type is not Organic) and (Contamination_Thickness is not High) and (Material_Type is Metal) and (Product_Shape is Simple) and (Cleaning_Process is Laser) and (Cleaning_Process_Time is Very_Fast) then (Cost is Low)(Cleaning_L
6. If (Contaminant_Type is not Organic) and (Contamination_Thickness is not High) and (Material_Type is Metal) and (Product_Shape is Complex) and (Cleaning_Process is Laser) and (Cleaning_Process_Time is Very_Fast) then (Cost is Medium)(Cleani
7. If (Contaminant_Type is not Organic) and (Contamination_Thickness is not High) and (Material_Type is Metal) and (Product_Shape is Very_Complex) and (Cleaning_Process is Laser) and (Cleaning_Process_Time is not Very_Fast) then (Cost is High)
8. If (Contaminant_Type is not Organic) and (Contamination_Thickness is not High) and (Material_Type is Metal) and (Product_Shape is Very_Complex) and (Cleaning_Process is Laser) and (Cleaning_Process_Time is Very_Fast) then (Cost is High)(Cle
9. If (Contaminant_Type is Organic) and (Contamination_Thickness is not High) and (Material_Type is Metal) and (Product_Shape is Simple) and (Cleaning_Process is Laser) and (Cleaning_Process_Time is not Very_Fast) then (Cost is Low)(Cleaning_L
10. If (Contaminant_Type is Organic) and (Contamination_Thickness is not High) and (Material_Type is Metal) and (Product_Shape is Simple) and (Cleaning_Process is Laser) and (Cleaning_Process_Time is Very_Fast) then (Cost is Low)(Cleaning_Lev
11. If (Contaminant_Type is Organic) and (Contamination_Thickness is not High) and (Material_Type is Metal) and (Product_Shape is Complex) and (Cleaning_Process is Laser) and (Cleaning_Process_Time is Very_Fast) then (Cost is Medium)(Cleaning
12. If (Contaminant_Type is Organic) and (Contamination_Thickness is not High) and (Material_Type is Metal) and (Product_Shape is Very_Complex) and (Cleaning_Process is Laser) and (Cleaning_Process_Time is not Very_Fast) then (Cost is High)(Cl
13. If (Contaminant_Type is Organic) and (Contamination_Thickness is not High) and (Material_Type is Metal) and (Product_Shape is Very_Complex) and (Cleaning_Process is Laser) and (Cleaning_Process_Time is Very_Fast) then (Cost is High)(Clean
14. If (Contaminant_Type is not Organic) and (Contamination_Thickness is High) and (Material_Type is Metal) and (Product_Shape is Simple) and (Cleaning_Process is Laser) and (Cleaning_Process_Time is not Very_Fast) then (Cost is Low)(Cleaning_
15. If (Contaminant_Type is not Organic) and (Contamination_Thickness is High) and (Material_Type is Metal) and (Product_Shape is Simple) and (Cleaning_Process is Laser) and (Cleaning_Process_Time is Very_Fast) then (Cost is Low)(Cleaning_Lev
16. If (Contaminant_Type is not Organic) and (Contamination_Thickness is High) and (Material_Type is Metal) and (Product_Shape is Complex) and (Cleaning_Process is Laser) and (Cleaning_Process_Time is not Very_Fast) then (Cost is Medium)(Clear
17. If (Contaminant_Type is not Organic) and (Contamination_Thickness is High) and (Material_Type is Metal) and (Product_Shape is Complex) and (Cleaning_Process is Laser) and (Cleaning_Process_Time is Very_Fast) then (Cost is Medium)(Cleaning
```

and	and	and	and	Then
Material_Type is	Product_Shape is	Cleaning_Process is	Cleaning_Process_Time is	Cost is
[✓ Metal]	[✓ Simple]	Non_Process	[✓ Very_Fast]	Very_Low
Non-Metal	Complex	[✓ Laser]	Fast	[✓ Low]
none	Very_Complex	Thermal	Medium	Medium
	none	Ultrasonic	Slow	High
		Abrasive	Very_Slow	Very_High
		Immersion	none	none
		Molten_Salt		
		Vibratory		
		none		
not	not	not	[✓ not]	not

Figure 12 – Fuzzy inference example

This toolbox it is set up only to predict the output of the performance of a process in terms of Cost, Cleaning Level and Energy consumption. Each cleaning process of the sequence is evaluated here and the results are used to determine if the sequence of process is suitable for the user or not. The suitability is determined by the constraints.

Software Flowchart

The software consists in four main sections including: User inputs and first seed generation, fuzzy sets and processes evaluation, stop conditions, and finally, crossover and mutation operations.

User Inputs and First Seed Generation

The user input, as explained before, are the initial conditions of piece that is being set up for cleaning. In this decision making model, four initial condition parameters are needed: contamination type, contamination thickness, material type and piece shape. Also, the user gives the inputs for the constraints. The constraints are used to evaluate if a process, or sequence of processes, meet the user need. The inputs used for the constraints are: total processing time allowed, minimum cleaning level needed and maximum energy consumption allowed. Any configuration that does not meet those constraints are disregarded and not counted as a feasible solution. Figure 13 and 14 gives a graphical view of this part of the software model. Also, the fuzzy toolbox file is read during this steps.

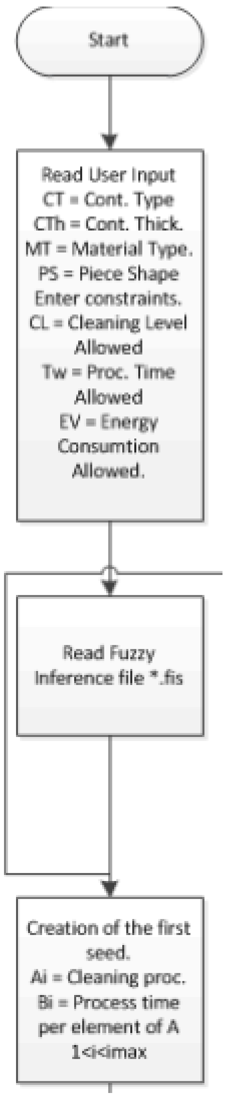
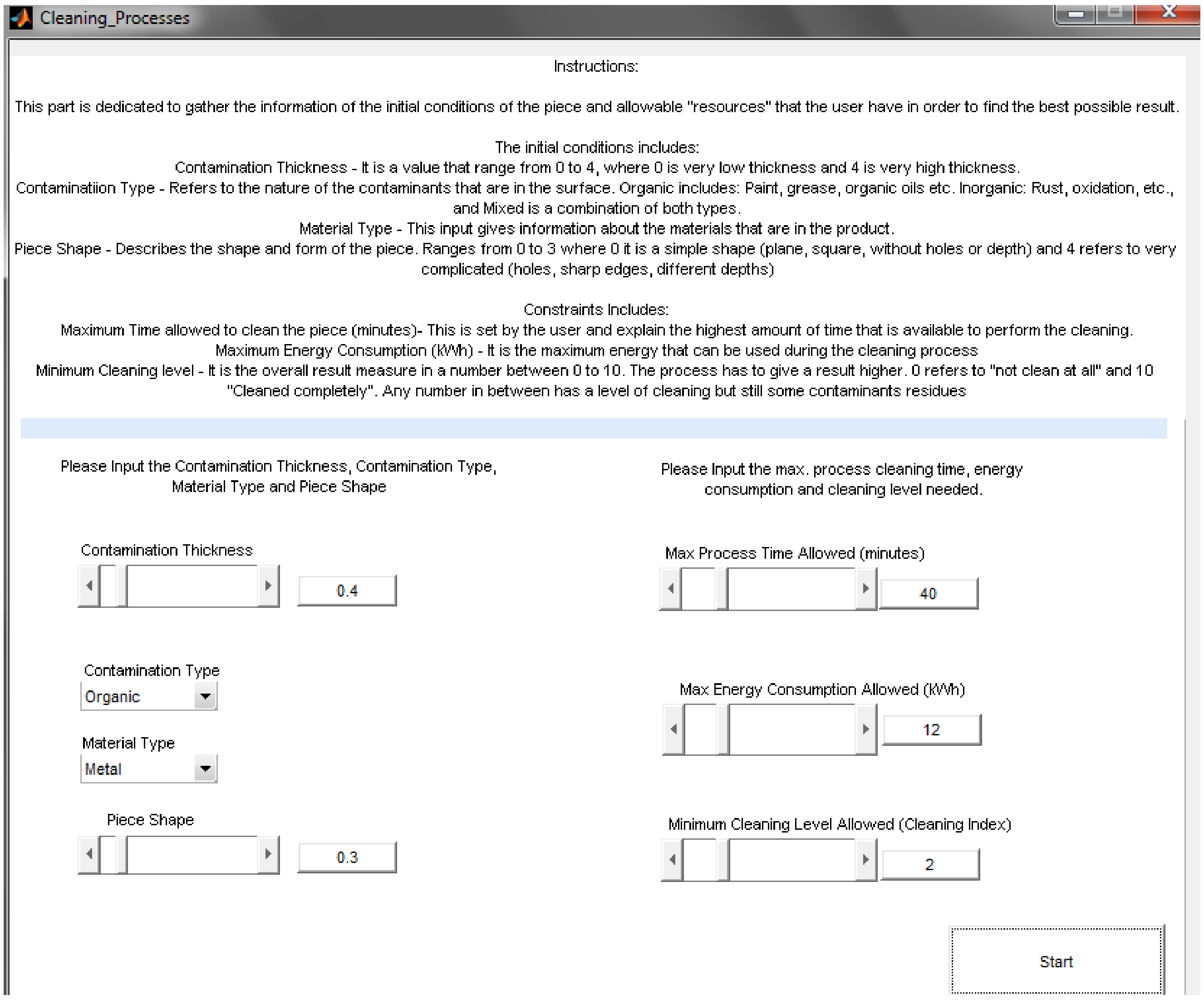


Figure 13 – User input and first seed generation


Cleaning_Processes

Instructions:

This part is dedicated to gather the information of the initial conditions of the piece and allowable "resources" that the user have in order to find the best possible result.

The initial conditions includes:
Contamination Thickness - It is a value that range from 0 to 4, where 0 is very low thickness and 4 is very high thickness.
Contaminatiion Type - Refers to the nature of the contaminants that are in the surface. Organic includes: Paint, grease, organic oils etc. Inorganic: Rust, oxidation, etc., and Mixed is a combination of both types.
Material Type - This input gives information about the materials that are in the product.
Piece Shape - Describes the shape and form of the piece. Ranges from 0 to 3 where 0 it is a simple shape (plane, square, without holes or depth) and 4 refers to very complicated (holes, sharp edges, different depths)

Constraints Includes:
Maximum Time allowed to clean the piece (minutes)- This is set by the user and explain the highest amount of time that is available to perform the cleaning.
Maximum Energy Consumption (kWh) - It is the maximum energy that can be used during the cleaning process
Minimum Cleaning level - It is the overall result measure in a number between 0 to 10. The process has to give a result higher. 0 refers to "not clean at all" and 10 "Cleaned completely". Any number in between has a level of cleaning but still some contaminants residues

Please Input the Contamination Thickness, Contamination Type, Material Type and Piece Shape

Contamination Thickness: 0.4

Contamination Type: Organic

Material Type: Metal

Piece Shape: 0.3

Please Input the max. process cleaning time, energy consumption and cleaning level needed.

Max Process Time Allowed (minutes): 40

Max Energy Consumption Allowed (kWh): 12

Minimum Cleaning Level Allowed (Cleaning Index): 2

Figure 14 – User input and constraints input GUI

Process Evaluation

The process evaluation consists of checking the seed generated to make sure that the sequence of cleaning processes associated with the seed meets the user requirements. This is performed by checking if they meet the constraints set by the user. Each element of the seed, with the results from the fuzzy sets, is compared with the processing time constraint, minimum cleaning level constraint and maximum energy consumption constraint. After the element is finished with the evaluation, the next element goes to the process. At the end, the ones that meet all the necessary constraints are kept as a feasible solutions. The elements that does not meet the constraints are disregarded. The checking is perform by a binary operation. The constraint that is met it assigned a number “1”. If not, a number “0” is assigned. When all three constraints are check, it begins a multiplication between the binaries number. An element that has all the number as “1” will be considered as feasible solution. Those which has a “0” are disregarded i.e.: Constraint check 1 x constraint check 2 x constraint check 3 = 1x1x1 = 1 = feasible solution. In other case, constraint check 1 x constraint check 2 x constraint check 3 = 1x0x1 = 0 = non feasible solution. Figure 15 shows the flow of this part of the software.

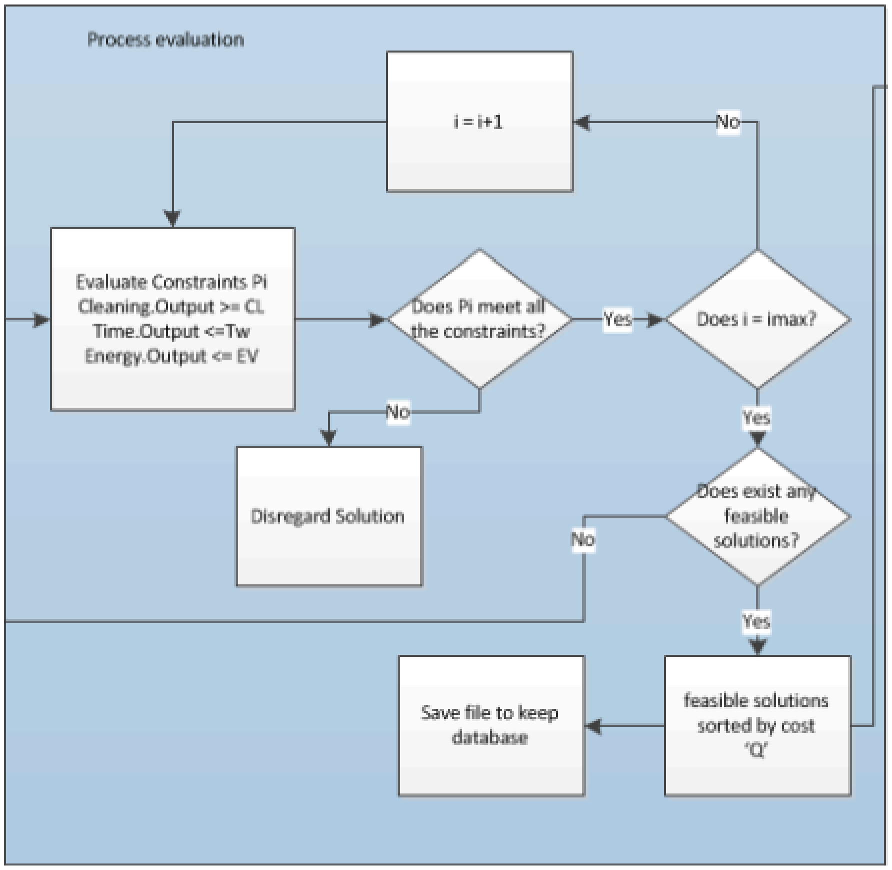


Figure 15 – Process evaluation loop

There are two options when there is no feasible solutions. 1) If the simulation has run for small amount of generations, then it should start over; 2) if the simulation has run for a specific number of generations and the user is satisfied, then the solutions should be used. At the end, it is the user who decides to run again the simulation or keep the solutions found.

Stop Condition

This part of the software is designed to stop simulation if the conditions are met. For this model, the stop conditions used depends in the number of generations performed and variance of the cost. The stop condition is set up to meet both requirements. In case that there is no feasible solutions during any generations, a display is shown to the let the user decide to run the simulation again or use the feasible solutions found. Figure 16 shows the process.

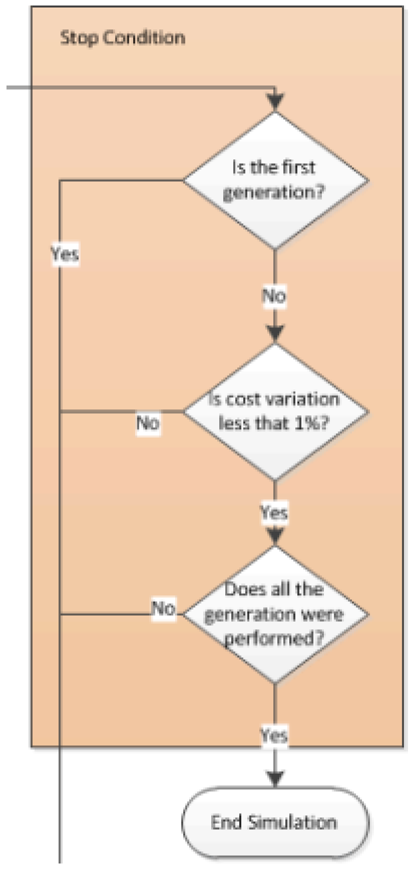
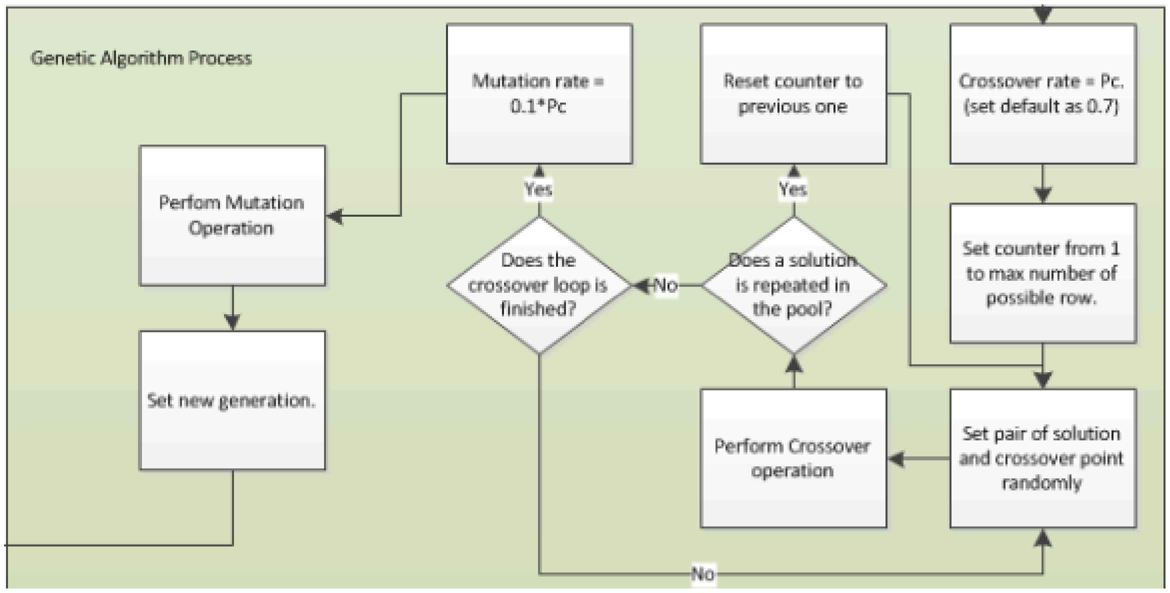

Stop Condition
Is the first generation?
Yes
No
Is cost variation less that 1%?
No
Yes
Does all the generation were performed?
No
Yes
End Simulation

Figure 16 – Stop condition flowchart

Genetic Algorithm

This part of the software intends to find better solutions every generation. The genetic algorithm mix the solutions founds in the current generation trying to capture the best characteristic of each array and put it together. From here, a new generation of possible solution is created and it is submitted to the fuzzy toolbox and the process evaluation. Figure 17 shows the genetic algorithm process.



Genetic Algorithm Process
Mutation rate = 0.1*Pc
Reset counter to previous one
Crossover rate = Pc, (set default as 0.7)
Perfom Mutation Operation
Yes
Yes
Does the crossover loop is finished?
No
Does a solution is repeated in the pool?
Set counter from 1 to max number of possible row.
Set new generation.
Perform Crossover operation
Set pair of solution and crossover point randomly
No

Figure 17 – Genetic Algorithm

The crossover rate is set to 0.7. Many articles and previous studies has shown that the crossover rate should be around 0.6 to 0.75 of the population. The mutation rate is set up to 10% of the population that perform a crossover. It is important to mention that the crossover point is done randomly. Sometimes it is done at one position, sometimes in other. This helps to simulate the “randomness” that exist in nature. Also, it helps to cover more possible solutions that may be a good fit. The genetic algorithm also checks if a solution is repeated in the same generation. In case that has happens, one of the element repeated is eliminated and another is created using crossover operation.

The user has the opportunity to decide how big is going to be each generation. This is to set the counter of the genetic algorithm when creating the elements of the next generation. If user decides that a population of 50 elements is wanted every generation, then the counter is set to 50. In case that a solution is repeated when creating the generation, the counter returns to the previous value. The checking process of repeated solution is performed after the second element of the generation is created and continues until the number of elements set by the user is met.

When performing the crossover operation, the software selects two elements of the solution, then set the crossover point randomly and creates the new element of the population. As it can be seen in figure 18, the process of the software is simple and straightforward:

1) User input and constraints
2) Generation of first seed. The size can be change inside the code.
3) Evaluate the element of the seed with the fuzzy toolbox to predict cost, energy consumption and cleaning level achieved.
4) Perform evaluation of the different elements of the array to check which ones meets the constraints.
5) If a certain number of generations are created, then enter to the stop condition loop to check if the cost variance is significant enough to stop the solution.
6) If the stop conditions have not triggered, go to the genetic algorithm process.
7) Perform the crossover and mutation operations. Check that a solution it is not repeated inside the same generation.
8) Set the new population of possible solutions.
9) Go to step 3 to begin the prediction of performance of the new population and evaluation. Stop when the stop conditions are met.

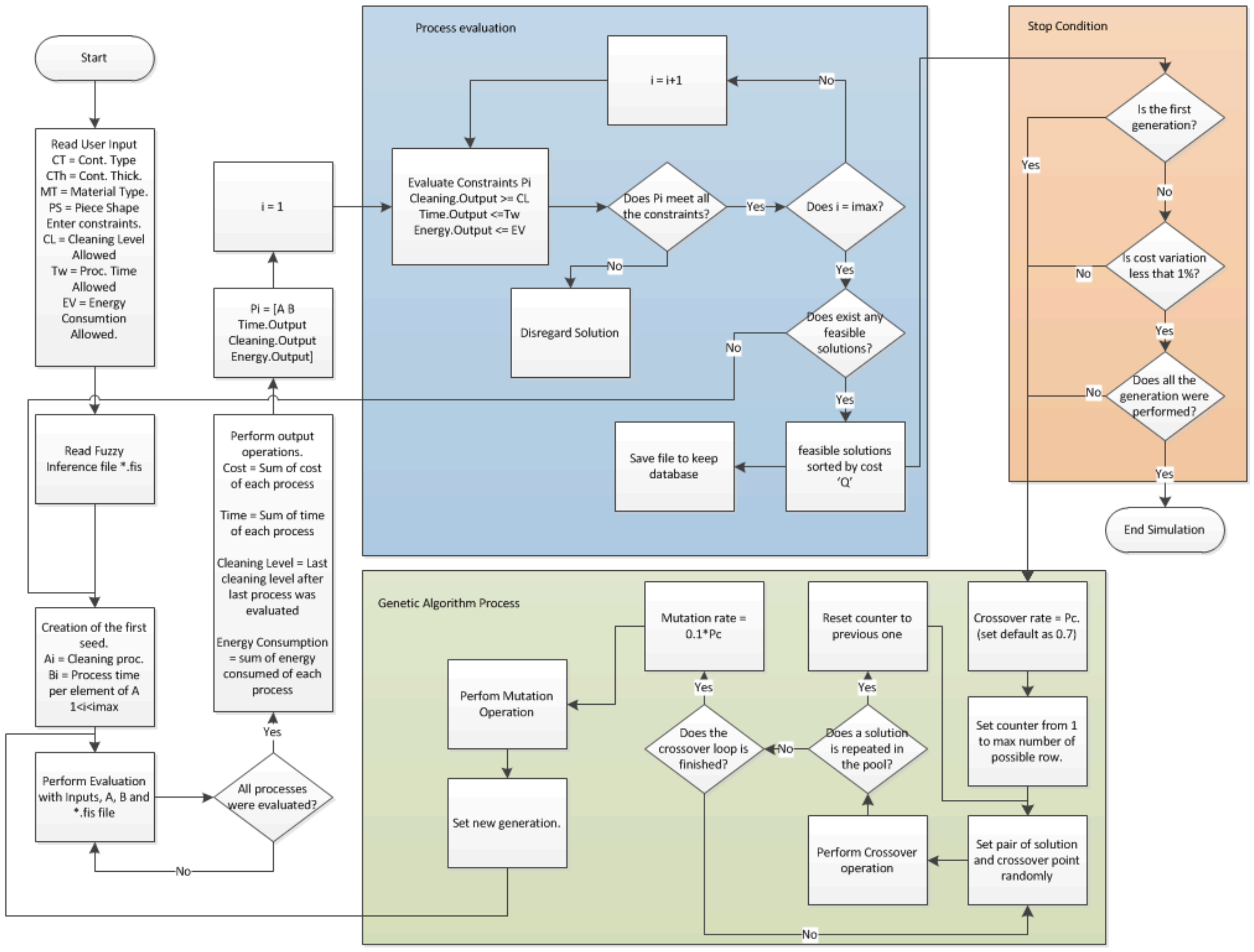

Start
Read User Input
CT = Cont. Type
CTh = Cont. Thick.
MT = Material Type.
PS = Piece Shape
Enter constraints.
CL = Cleaning Level Allowed
Tw = Proc. Time Allowed
EV = Energy Consumtion Allowed.
i = 1
Pi = [A B Time.Output Cleaning.Output Energy.Output]
Read Fuzzy Inference file *.fis
Perform output operations.
Cost = Sum of cost of each process
Time = Sum of time of each process
Cleaning Level = Last cleaning level after last process was evaluated
Energy Consumption = sum of energy consumed of each process
Creation of the first seed.
Ai = Cleaning proc.
Bi = Process time per element of A
1<i<imax
Perform Evaluation with Inputs, A, B and *.fis file
All processes were evaluated?
Yes
No
Process evaluation
i = i+1
Evaluate Constraints Pi
Cleaning.Output >= CL
Time.Output <=Tw
Energy.Output <= EV
Does Pi meet all the constraints?
Yes
No
Does i = imax?
Disregard Solution
Does exist any feasible solutions?
feasible solutions sorted by cost 'Q'
Save file to keep database
Stop Condition
Is the first generation?
Is cost variation less that 1%?
Does all the generation were performed?
End Simulation
Genetic Algorithm Process
Mutation rate = 0.1*Pc
Reset counter to previous one
Crossover rate = Pc. (set default as 0.7)
Perfom Mutation Operation
Does the crossover loop is finished?
Does a solution is repeated in the pool?
Set counter from 1 to max number of possible row.
Set new generation.
Perform Crossover operation
Set pair of solution and crossover point randomly

Figure 18 – Software Flowchart

Case Studies

Two cases are shown to demonstrate that the software works and gives the desired results. The case are numbered as case 1 and case 2. Different inputs are used in each case. At the end of each case, the best solutions of each generation are shown with a plot of the cost.

The following information tables are used for all the cases:

Table 1 - Description of contamination types

Contamination Type	Description
1	Organic only. (Oil, grease, organic paints, etc.)
2	Inorganic only (oxidation, rust, dust, etc)
3	Mixed contaminations - Organic and Inorganic

Table 2 - Description of contamination thickness

Contamination Thickness	Description
0	Contamination cannot be seen by the human eye
1	Very Thin Layers of contaminants
2	Thin layers of contaminants
3	Medium layer of contaminants
4	High Presence of contaminants. Very thick layers

Table 3 - Description of material type

Material Type	Description
1	Metal
2	Non - Metal

Table 4 - Description of piece shape index

Piece Shape Index	Description
0	Very simple shape i.e.: sheet of metal, plain.
1	Some presence of complicated part i.e: curves, depth, spikes
2	Presence of complicated shapes and holes
3	Very complicated shape - Many holes, spikes, lack of support.

Table 5 - Cleaning Level Index

Cleaning level Index	Description
0	High presence of contaminants
1	Super Minor removal of contaminants
2	Minor removal of contaminants
3	Minor Cleaning - Presence of cleaned spots starts to appear.
4	Minor Cleaning - Small Presence of cleaned spots
5	Medium Cleaning - Still many contaminants
6	Medium Cleaning - Spots with contaminants
7	Good cleaning - Some major spot with contaminants
8	Good Cleaning - Still some spots with contaminants
9	Excellent Cleaning - Contaminant cannot be seen by human eye
10	Excellent cleaning - No contamination at any level

For tables 2, 4 and 5, the index is a “real” number, may have decimals. For example, an index of 0.5 in the contamination thickness, may refer to a very thin layers that are not straightforward to see but also are kind of seeable for the human eye. This is why the index cannot be a zero or 1, it is a number in between. For tables 1 and 3, the index are the one shown in the table and cannot have any type of decimals. These index are set as a “Boolean” index (one or the other).

Case 1:

Table 6 - Case 1 Input

Inputs	Index	Description
Contamination type	1	Organic
Contamination Thickness	3.4	High
Material Type	1	Metal
Piece Shape	2.1	Complex to Very Complex
Cleaning Level Constraint	9.2	Minimum Allowed
Energy Constraint	20	Max. Allowed
Time Constraint (min)	80	Max. Allowed

Table 7 - Case 1 Best solutions

Generation	Cleaning Process 1	Cleaning Process 2	Cleaning Process 3	Cleaning Process 4
1	Vibratory	■	Thermal	Immersion
2	■	Thermal	Vibratory	Immersion
3	■	Thermal	Immersion	■
4	■	Thermal	■	Immersion
5	■	Immersion	Vibratory	■
6	■	Immersion	Immersion	■
7	■	Molten Salt	Immersion	■
8	■	Immersion	■	■
9	■	■	Immersion	■
10	■	■	Immersion	■
11	■	■	Immersion	■
12	■	■	Immersion	■
13	■	■	Immersion	■
14	■	Immersion	■	■
15	■	■	■	Immersion
16	■	■	■	Immersion
17	■	■	■	Immersion
18	■	■	Immersion	■
19	■	■	Immersion	■
20	■	■	■	Immersion

Table 8 - Case 1 Results

Generation	Cost	Total Time(min)	Cleaning level	Energy Consumption(Kwh)
1	$107,894.91	65	9.45	11.9
2	$107,894.91	51	9.45	8.8
3	$71,929.94	58	9.45	11.44
4	$71,929.94	21	9.45	5.49
5	$71,929.94	31	9.45	7.17
6	$71,929.94	39	9.45	11.31
7	$71,929.94	41	9.45	9
8	$35,964.97	26	9.45	6.5
9	$35,964.97	20	9.45	6.5
10	$35,964.97	25	9.45	6.5
11	$35,964.97	25	9.45	6.5
12	$35,964.97	25	9.45	6.5
13	$35,964.97	25	9.45	6.5
14	$35,964.97	26	9.45	6.5
15	$35,964.97	26	9.45	6.5
16	$35,964.97	26	9.45	6.5
17	$35,964.97	26	9.45	6.5
18	$35,964.97	25	9.45	6.5
19	$35,964.97	25	9.45	6.5
20	$35,964.97	26	9.45	6.5

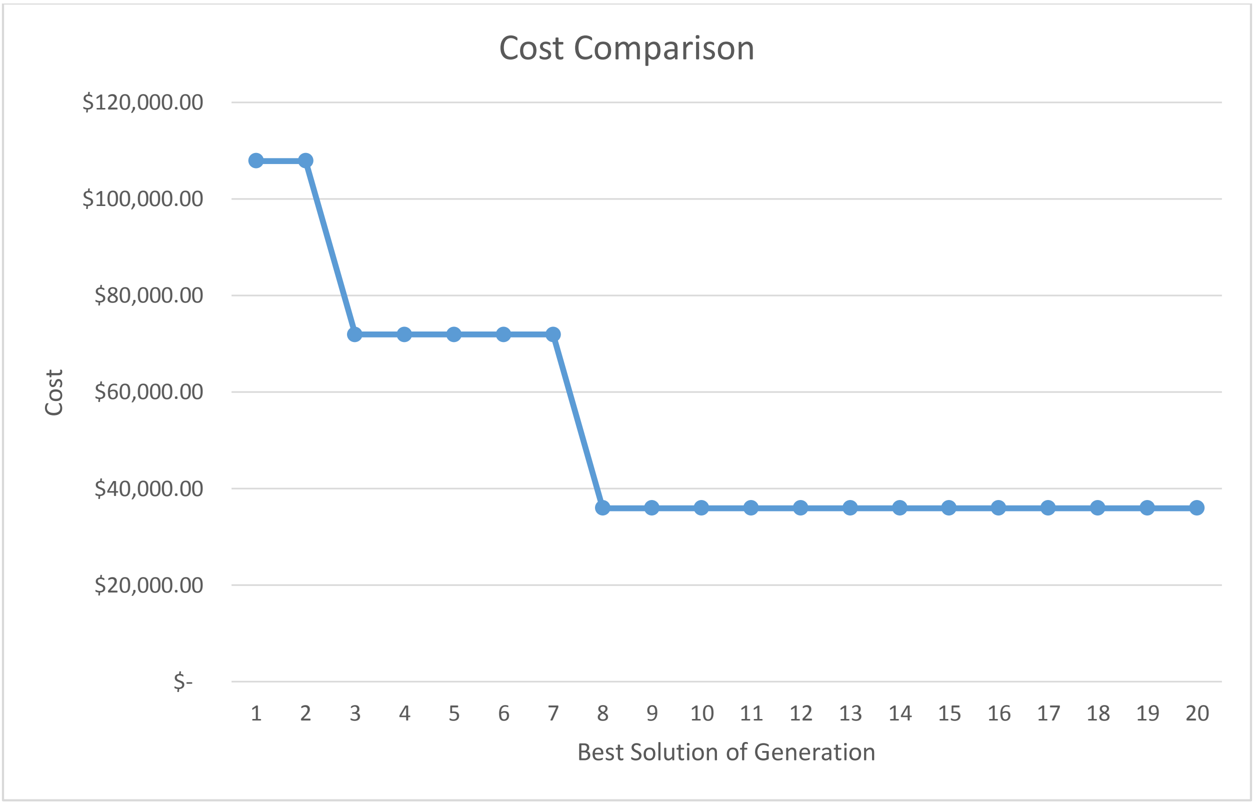


Figure 19 – Case 1 Cost plot of best solutions

From the plot, the best solution is located up to generation 8 until generation 20. The cleaning process that reached the best solutions is Immersion Cleaning for 25 minutes, giving a cleaning level of 9.45 (excellent cleaning) and energy consumption of 6.50 kWh. The system cost predicted is $35,964.97. Taking in consideration the inputs provided: High levels of contamination, a complex to very complex shape plus organic contamination, there is no surprises that immersion is a method that dominates in this kind of conditions. Immersion cleaning has a high impact on organic contamination and also supports high level of contamination thickness and any morphology of the piece. This is because the piece that is set for the immersion can be cleaned because the morphology does not play an important role. Immersion cleaning may reach tight spaces that with other cleaning method may not be possible. Even though Abrasive, Laser and molten salt may be suitable, there are flaws that Immersion cleaning does not. Abrasive and laser can deal with organic contamination and thick layers of contamination. But the piece shape plays an important role on this cleaning method. These two cannot support very complicated shapes because it won’t clean the piece as it should be. The flaw of Molten Salt cleaning is that it takes longer time to achieve the cleaning level desired and may be more expensive. This is why this process does not appear as one of the best one.

As it can be seen, in the case study provides an insight of how software works and gives logic answers. The user can modify the parameters and run the program again to have an idea of which cleaning process should use depending on their need.

Case 2

Table 9 - Case 2 Input

Inputs	Index	Description
Contamination type	3	Mixed
Contamination Thickness	2.52	High
Material Type	1	Metal
Piece Shape	1.8	Complex
Cleaning Level Constraint	8.5	Minimum Allowed
Energy Constraint	18	Max. Allowed
Time Constraint (min)	80	Max. Allowed

Table 10 - Case 2 Best Solutions

Generation	Cleaning Process 1	Cleaning Process 2	Cleaning Process 3	Cleaning Process 4
1	Immersion	Vibratory	Molten Salt	
2		Immersion	Molten Salt	
3	Abrasive	Molten Salt		
4	Vibratory	Abrasive		
5	Vibratory		Abrasive	
6	Vibratory	Abrasive		
7	Vibratory			
8		Vibratory		Abrasive
9	Vibratory			
10		Vibratory		Abrasive
11		Vibratory		Abrasive
12	Vibratory		Abrasive	
13	Vibratory			
14	Vibratory			
15	Vibratory			
16		Vibratory	Abrasive	
17	Vibratory		Abrasive	
18	Vibratory			
19	Vibratory			
20	Vibratory			

Table 11 - Case 2 Results

Generation	Cost	Total Time(min)	Cleaning level	Energy Consumption(Kwh)
1	$ 107,894.91	73.00	9.45	14.12
2	$ 71,929.94	88.00	9.45	15.44
3	$ 71,929.94	39.00	9.28	9.00
4	$ 71,929.94	36.00	9.30	7.17
5	$ 71,929.94	41.00	8.81	7.00
6	$ 71,929.94	25.00	9.28	7.17
7	$ 35,964.97	36.00	9.37	2.50
8	$ 71,929.94	25.00	9.45	7.17
9	$ 35,964.97	36.00	9.37	2.50
10	$ 71,929.94	35.00	9.45	7.00
11	$ 71,929.94	35.00	9.45	7.00
12	$ 71,929.94	35.00	8.98	7.05
13	$ 35,964.97	36.00	9.37	2.50
14	$ 35,964.97	36.00	9.37	2.50
15	$ 35,964.97	36.00	9.37	2.50
16	$ 71,929.94	51.00	9.45	7.00
17	$ 71,929.94	51.00	9.45	7.00
18	$ 35,964.97	36.00	9.37	2.50
19	$ 35,964.97	36.00	9.37	2.50
20	$ 35,964.97	36.00	9.37	2.50

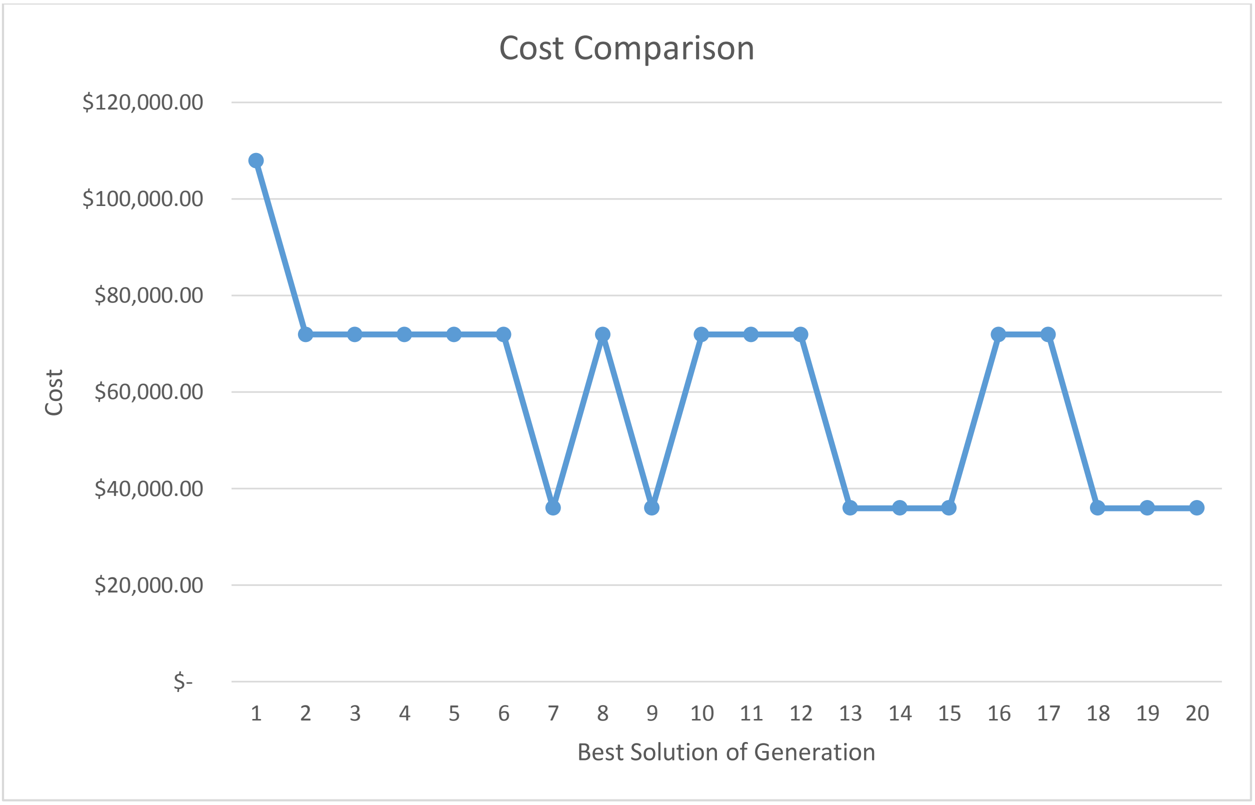


Figure 20 – Case 2 Cost plot of best solutions

From the plot, the best solution are located in generations 7, 9, 13 to 15 and 18 to 20. The cleaning process that reached the best solutions is Vibratory, giving a cleaning level of 9.37 (excellent cleaning) and energy consumption of 2.50 kWh. The system cost predicted is $35,964.97. From the plot, there are some spikes where the cost is increased. The reason of this is that the method used to search for the best solution is indeed a heuristic approach. It is not entirely perfect and some solutions may be lost when performing the crossover and mutation operations. Also, since the fuzzy inference is used to predict the performance of the solutions, variations are expected due to the characteristic of the fuzziness.

In both cases, the cost is being reduced during the simulation. The solution may be repeated in terms of cleaning processes chosen, but changes the processing time and cleaning level. The comparison is not only done between the different processes, but also different processing time of a same cleaning process. The method it is not entirely perfect, but it works to give the user a better idea of which cleaning processes should use when performing the process planning.

CHAPTER V

CONCLUSIONS AND FUTURE RESEARCH

The two case studies shown before demonstrates that the mix of fuzzy sets and genetic algorithm is a powerful tool to make decisions for process planning with uncertain and imprecise information. It is important to mention that even though the information of the cleaning processes are based on literature review and is not entirely exact, it is possible to reach a solution that is not far from reality.

It shall be noted that the range given may change depending on the size of the batches and the quality level of the equipment. More accurate and bigger equipment may also be more expensive. The cases are also focused on a completely new process that is going to be implemented. In case that the remanufacturer has some cleaning processes established, or other methods not included in this study, the user still can easily include those processes in the fuzzy toolbox and eliminate those that are not part of their process. For example, if a user has only three processes such as: Molten Salt, Thermal and Abrasive with dry sponge, the fuzzy sets should be modified in order to gather more accurate answers. All the user need is to include information about the cost, cleaning level and energy consumption for each case in terms of contamination thickness, material types, shape of the piece, and contamination type. In case of having own cleaning processes the user can also input all the information to the fuzzy sets.

A very special issue is that different type of user has different needs. Some user has its own measures and may feel comfortable with its own parameters depending on the size of their process. Maybe a Low cost for one user may range from $500 to $2,500, while to another user it may start at $30,000. In this case, the first user has a smaller process than the second one. This is why each user has to modify their membership functions depending on their size of their process.

In conclusion, the methodology proposed in the research fulfilled its purpose. Even with new cleaning processes, the model can be easily updated to incorporate the new ones such as Ultrasonic Piezoelectric and Magnetostrictive, Abrasive Wet, Dry and with dry sponges, and molten salt with different kind of mixes. Since there is no overall process that cleans all type of contaminants effectively, the methodology develop in this research provided significant contribution because it allows the user compare between different options of cleaning processes with different inputs. This leads to a better decision-making process. Specifically, the user or remanufacturer can simulate the introduction of a new cleaning method. In this case, the user can improve their process by comparing this new cleaning method with his existing ones and simulate how advantageous may be to implement it as part of their process or not. This is done by including the process in the software and run the simulations. If the process is chosen because it delivers a good cost with good cleaning level, then the process should be included.

In addition, the software proposed intends to give an initial step in the decision making process for cleaning processes in remanufacturing. As an essential part of the process planning, it is important to make the best possible decision in the planning process to avoid unexpected changes that will increase the cost. But it becomes very hard to make a decision when the variables, parameters and conditions are not as exact as wanted to be. There is high uncertainty that makes the decision process very difficult. With the methodology proposed and the fuzzy inference based expert systems, a decision can be made for all similar occasions.

There are, however, limitations to the models developed in this research. Future research is suggested to overcome these uncovered areas. This approach has many variables not considered but may be important for the remanufacturing process. Batch sizes, measure of contamination layers, sizes of the cleaning equipment, quantity of worker and other important variables are not taken in consideration in this model. Many more parameters may be included in this study, for example, the surface finish. Some cleaning processes are not as incisive as other. An abrasive method may leave a bad surface finish. On the other hand, Ultrasonic leaves an untouched surface finish. This at the end dictates how much reprocessing of the piece would need after the cleaning process. Similarly, the hardness or material composition may change with processes that work at high temperatures. The material condition is an important measure that have to be included as part of the output parameters.

The user may also need to modify the decision-making parameters. Instead of having the comparison only based on the cost, the user can create utility function to choose the solutions that gives higher satisfaction. Sometimes, the cost is not the only important process performance measures, overall cleaning level may be important too. By setting the utility function that considers both cost and cleaning level, the user could obtain a solution that bring better user satisfaction.

REFERENCES

ACOSTA, G.; TODOROVICH, E.; 2011 “Genetic algorithms and fuzzy control: a practical synergism for industrial applications”. Consejo Nacional de Investigaciones Científicas y Técnicas and Universidad Nacional del Centro de la Prov. de Buenos Aires. Computers in Industry. ELSevier, Vol. 52, pp. 183 – 195.

AWAD S.B., NAGARAJAN R. “Developments in Surface Contamination and Cleaning Chapter-6 - Ultrasonic Cleaning”. Crest Ultrasonic Corporation, Trenton, NJ, USA. Indian Institute of Technology Madras, Chennai, India

BARBA, D., OLIVEIRA, J.,SALIS, J., SCHUCH, C., 2013 “Remanufacturing versus Manufacturing – Analysis of Requirements and Constraints for a Study Case: Control Arm of a Suspension System” Instituto Federal de Educação, Ciência e Tecnologia Sul Riograndense, Campus Sapucaia do Sul, Centro de Competência em Manufatura, Brazil, 20th CIRP International Conference on Life Cycle Engineering, Singapore, 2013 pp.: 679, 773

BAZARAA, M.; SHERALI, H. 2006 “Non Linear Programming: Theory and Algorithms” 3rd edition. Publisher: John Wiley & Sons, pp. 1-4.

BROWN, M., HARRIS, C. 1994, “Neurofuzzy Adaptive Modeling and Control” Prentice Hall International (UK). Taken from http://code.eng.buffalo.edu/tracking/Fuzzy/main.htm March 26, 2014

CHEN, G.; PHAM, T. 2000 “Fuzzy Sets, Fuzzy Logic and Fuzzy Control Systems” 1st. edition. Publisher: CRC Press LLC, 2001.

CHUAN SHENG SI. On Self-diagnosis Technique of Wear in Auto Cylinder-Piston System Faculty of Transportation Engineering Huaiyin Institute of Technology Huai an, China

CIAMPINI, D.“Impact Velocity, Almen Strip Curvature and Residual Stress Modeling in Vibratory Finishing” Department of Mechanical and Industrial Engineering University of Toronto © Copyright by David Ciampini (2008)

DAVIS, T. 1993 “Effective supply chain management, Sloan Management Review” pp 35–46.

EL HALABI, E. and DOOLAN, M. 2013 “Operational Challenges in the Automotive Recycling Business: A System Dynamics Perspective” The Australian National University. 20th CIRP International Conference on Life Cycle Engineering, Singapore, 2013, pp. 353 – 355.

FRENZEL, L. “A Comparison Of Surface Preparation For Coatings By Water Jetting And Abrasive Blasting”. Advisory Council San Marcos, Texas, U.S.A. Prepared and Presented at 1999 WJTA Conference; Houston TX. Printed in the Proceedings.

HAMMONR, R (1998) “Issues in the Automotive Parts Remanufacturing Industry – A Discussion of Results from Surveys Performed among Remanufacturers”, Geogia Institute of Technolgy, Atlanta, Georgia, pp.5

JUNIOR, M.L, FIHLO,M.G., 2011. “Production planning and control for remanufacturing: literature review and analysis.” Production Planning & Control.

KNOTH, G., JACKSON, D., SORBO, N.“Automated CO2 Composite Spray Cleaning System for HDD Rework Parts”, Cool Clean Technologies, Inc., PurCO2 Division, 2006

KUMARA,D.; SINGH. J.; SINGH, O., SEEMA, 2010 “A fuzzy logic based decision support system for evaluation of suppliers in supply chain management practices” Beant College of Engineering and Techonology. ELSevier. Mathematical and Computing Modeling, Vol. 57, pp. 2945 - 2955

LONG, Y.Y., and LI, J., 2013“Modeling and optimization of the molten salt cleaning process” University of Texas – Pan American. ElSevier, pp.:

MEHAN, R. “The wear of selected materials in mineral oil containing a solid contaminant” General Electric Company, Corporate Research and Development Center, Schenectady, NY 12301 (U.S.A.) (Received October 28,1987; accepted December 14, 1987)

MOHEBBI, A.; TAHERI, M.; NOOSHIRAVANI A. 2011 “Modeling of thermodynamics properties of refrigerants using artificial neural networks and genetic algorithm” First edition, Novinka Science Publishers, pp. 10 – 11.

NATIONAL AERONAUTICS and SPACE ADMINISTRATION (NASA), “Contamination Control Handbook”, SP-5076, 1969, Washington, D.C.

NATIONAL CENTER FOR REMANUFACTURING & RESOURCE RECOVERY (NC3R) – Ultra Sonic Cleaning of Automotive Parts

PEIDRO, D.; MULA, J.; POLER, R., 2009 “Fuzzy optimization for supply chain planning under supply, demand and process uncertainties” Universidad Politecnica de Valencia and Universidad de Granada. ElSevier. Fuzzy sets and Systems. Vol. 160. Pp. 2640 – 2657

PHILLIPS, C.; KARR, C. and WALKER, G. “Helicopter Flight Control with Fuzzy Logic and Genetic Algorithms”, Logic Techonologies, University of Alabama and Aerobotics Corp. pp. 1 &13.

RAJANGAM, K., ARUNACHALAM, V., SIBRAMANIA, R., “Fuzzy Logic Controlled Genetic Algorithm to solve the economic load dispatch for the thermal power station”. European Scientific Journal, Vol. 8, No. 7, pp. 172 & 183.
SHANSAN, Y., S.K. Ong, and A.Y.C. Nee 2013, “Design for Remanufacturing – A Fuzzy-QFD Approach”, National University of Singapore, pp.1

STEINHILPER, R. (1995) “REMANUFACTURING: The ultimate form of recycling” Germany, pp.7, 28

TAM, A., PARK, H., COSTAS, P. “Laser cleaning of surface contaminants”. IBM Almaden Research Center, IBM Manufacturing Technology Center. Department of Mechanical Engineering, University of California.

TANAKA, S., MATSUOKA, Y. NAKAMURA, H. “Thermal cleaning and growth temperature effects on deep levels of Be-doped p-InAlAs grown on InP by molecular beam epitaxy”. Central Research Laboratory, Hitachi, Ltd. Kokubunji, Tokyo 185, Japan

SHOEMAKER, R.H. “Salt Bath Cleaning Process”. Kolene Innovation in Surface Technologies 2003 Kolene Corporation Research Results

TIWARI, M.; HARDING, J.; 2011 “Evolutionary computing in advanced manufacturing” First edition, Scrivener Publishing and John Wiley & Sons, pp. 10-14

WEIWEI, L., BIN, Z., MING, Z., YANGZEN, L. and HONG-CHAO, Z. “Study on Remanufacturing Cleaning Technology in Mechanical Equipment. Remanufacturing Process” Dalian University of Techonology and Texas Tech University. 20th CIRP International Conference on Life Cycle Engineering, Singapore, 2013, pp. 1 and 643, 647 – 648.

ZHANG, H., JUNZHONG, Z. “Study on the Safety Design Method of Ultrasonic Cleaning for the Motor winding”. Institute of Electromechanical Equipment, Qingdao Navy Submarine Academe, Qingdao, China

YAGAR, O. 2012 “Remanufacturing cleaning process evaluation, comparison and planning”, University of Texas Pan-American. Pp 18 – 81, 122 – 125.

APPENDIX

APPENDIX A

MATLAB FUZZY INFERENCE PROGRAMMING

```
function Q = Thesis_Program_Cleaning_Processes(CT,CTh,MT,PS,Tw,CLp,Ev)

% Reads the Fuzzy Rules from Matlab Toolbox
name = uigetfile('Fuzzy_Memberships_and_Rules_New.fis');
fisrmat = readfis(name);

%% Random Generation Seed

% Size of the first population
h=1800;

% Sequence of Cleaning Process
for i=1:h
    temp = randperm(8);
    a(i,:) = temp(1:4)-1;
end

% Processing time of each cleaning process
b = unidrnd(60,h,4);

% Same user input for all generations
c(1:h,1) = CT;     % Contamination type Input
c(1:h,2) = CTh;    % Contamination thickness Input
c(1:h,3) = MT;     % Material type Input
c(1:h,4) = PS;     % Piece Shape Input

%% Starts the Generations
pp = 10;          % Refers to the number of generations.
for perm = 1:pp
% Information matrix, c=inputs, a=cleaning processes,% b=cleaning time
U=[c,a,b];
v=U;
l = size(v);

%% Performs the fuzzy evaluation
```

```
for k = 1:4
    for q = 1:l(1)
        % Result of each cleaning process after performing the Fuzzy Set
        % Rules
        output(q,k*4-3:k*4) = evalfis(v(q,[1:4 k+4 k+8]),fismat);
% Refresh the New contamination Thickness to be used as new input
% for the next cleaning process inside the sequence.
    v(q,2) = output(q,k*4);
    end
end

% Binary Matrix for cleaning process (If there is non-process
% then y(i,j)= 0, any othe process y(i,j) = 1)
y = a>0;

%% Sequence Outputs
 % Sumation of Cost - Ouput
g = y(:,1).*output(:,1)+y(:,2).*output(:,5)+y(:,3).*output(:,9) ...
+y(:,4).*output(:,13);

 % Sumation of Energy Consumption - Output
z = y(:,1).*output(:,3)+y(:,2).*output(:,7)+y(:,3).*output(:,11)...
+y(:,4).*output(:,15);

 % Cleaning Level Output
temp = output(:,[2 6 10 14]);
r = max(temp,[],2);

 % Sumation of Cleaning Time - Output
s = sum(y.*v(:,9:12),2);

 % Matrix after fuzzy evaluation - Includes outputs
P = [U g s r z];

%% Solutions evaluations (1 = satisfies constraint, 0 = not satisfies)
ii = z <= Ev;                       % Energy Consumption Constraint
iii = s <= Tw;                      % Process Time Constraint
jj = r >= CLp;                      % Clean Level Constraint
ij = logical(ii .* iii .*jj);       % Checks if all constraints are met

% Feasible Solutions - Disregard solutions that does not meet the
% constraints
Q = P(ij,:)

% When no solutions are found, display the message. Either run the program
% again, or check the results found.
if isempty(Q)&&perm >= 7
        msgbox('No more solutions found');
    else
        if isempty(Q)&&perm<=7
             msgbox(['Not enough solutions - Try again. Generations = '...
            ,num2str(perm)])
        end
end

% Sort rows by cost. Best cost is put as the first one.
gh = size(Q);
Qnew = sortrows(Q,13);
Q=Qnew;

% Matrix of best solutions of each generation.
i = perm;
    for j = 1:16
        optimal_solution(i,j) = Q(1,j);
    end

% Plots the best solution found in each generation.
f = optimal_solution(:,13);
figure(1);
plot(f,'-.r*','MarkerEdgeColor','k');
hold on;
grid on;
title('Best Solutions');   % Title of the plot
xlabel('Generation');      % Name of axis 'x'
ylabel('Cost(USD)');       % Name of axis 'y'

% Saves Feasible Solutions in a matlab file '.m'
filename =['Feasible_Solutions_', num2str(perm), '.mat'];
delete(filename);
save(filename);

% Save Feasible Solutions in a .xls file (Excel)
filename_1=['Trial','.xls'];
Generation = num2str(perm)
xlswrite(filename_1,Q,Generation);
filename_2=['Best_Solutions','.xls'];
xlswrite(filename_2,optimal_solution,Generation);

%% Stop condition. If cost has not change in more than 1% and has reached a
% certain number of generations.
for i = 3:perm
            stop_condition = abs((optimal_solution (i-1,13)-...
                optimal_solution(i,13))/optimal_solution(i,13));
            if stop_condition <= 0.01 && perm>=8
                jjj = msgbox('Stop Condition Met');
                display (jjj);
                return;
            end
end


%% Genetic Algorithm
```

```
% Crossover rate for feasible solutions.
PC = 0.7;

% Round the result to avoid decimals in the croossover total number
tt = round(PC*gh(1));
cd_1= randi(16);             % Random Case for crossover point
cd = cd_1;
hh = 50;
for lk = 1:hh                % Creates the offspring given a case
    xx = randi(tt,1,1);
    yy = randi(tt,1,1);
    switch cd
        case 1
            temp3 = [yy yy yy yy];
        case 2
            temp3 = [yy yy yy xx];
        case 3
            temp3 = [yy yy xx yy];
        case 4
            temp3 = [yy yy xx xx];
        case 5
            temp3 = [yy xx yy yy];
        case 6
            temp3 = [yy xx yy xx];
        case 7
            temp3 = [yy xx xx yy];
        case 8
            temp3 = [yy xx xx xx];
        case 9
            temp3 = [xx yy yy yy];
        case 10
            temp3 = [xx yy yy xx];
        case 11
            temp3 = [xx yy xx yy];
        case 12
            temp3 = [xx yy xx xx];
        case 13
            temp3 = [xx xx yy yy];
        case 14
            temp3 = [xx xx yy xx];
        case 15
            temp3 = [xx xx xx yy];
        case 16
            temp3 = [xx xx xx xx];
    end

    tempsec = [Q(temp3(1),5) Q(temp3(2),6) Q(temp3(3),7)...
        Q(temp3(4),8)];

% Checks that a sequence of processes are not repeated in the solution.
    if lk == 1
```

```
        Ua(lk,:) = tempsec;
    else
        ss = size(Ua);
        for uq = 1:ss(1)
        if tempsec == Ua(ss(1),:);
         lk = lk-1;

        else
            Ua(lk,:) = tempsec;
        end
        end
    end

end



hh = size(Ua);
ib = randi(gh(1),hh(1),4);
Qb = Q(:,9:12);
Ub = Qb(ib);

Uc(1:hh(1),1) = CT;
Uc(1:hh(1),2) = CTh;
Uc(1:hh(1),3) = MT;
Uc(1:hh(1),4) = PS;

% Mutation Operations
Pm = 0.1;
for i = 1:round(lk*Pm)
    temp = randperm(8);
    ap = temp(1)-1;
    g = randi(4,1,1);
    h = randi(lk,1,1);
% Of possible answers, change a value of a = Cleaning Processes.
    Ua(h,g) = ap;
end

a = Ua;     % Matrix of cleaning processes after crossover and mutation
b = Ub;     % Matrix of processing time after crossover and mutation
c = Uc;     % Matrix of input - Same as initial.

 % Information matrix ready for evaluation for next iteration.
Qnew = [Uc Ua Ub];
U = Qnew;

clear Q;
clear P;
clear g;
clear z;
clear r;
clear s;
clear v;
clear output;
clear y;
clear cd;
clear Qnew;
perm;

end
end
```

APPENDIX B

APPENDIX B

MATLAB® FUZZY INFERENCE RULES

Matlab® gathers the information of the fuzzy inference in this format. For the fuzzy rules, it is not shown the linguistic description. In fact, there are only numbers. The explanation of the rules consist in the following:

The first six columns refers to the input (CT, CTh, MT, PS, y_j, t_j). The number of each one refers to the subset used. For example, [1 1 1 1 4 3] refers to [Organic Low Metal Simple Ultrasonic Medium]. The next fourth columns are the subset of the outputs [Cost Cleanliness_Level Energy_Consumption New_Cont._Thickness]. The number in parenthesis refers to the weight of rule and the last number is the "AND" concatenation (set as 1).

CT – 1 = Organic, 2 = Inorganic, 3 = Mixed, 0 = None

CTh – 1 = Low, 2 = Medium, 3 = High, 0 = None

MT – 1 = Metal, 2 = Non-Metal, 0 = None

PS – 1 = Simple, 2 = Complex, 3 = Very Complx, 0 = None

y_j – 1 = No Process, 2 = Laser, 3 = Thermal, 4 = Ultrasonic, 5 = Abrasive, 6 = Immersion, 7 = Molten Salt, 8 = Vibratory

t_j – 1 = Very fast, 2 = Fast, 3 = Medium, 4 = Slow, 5 = Very Slow, 0 = None

Cost – 1 = Very Low, 2 = Low, 3 = Medium, 4 = High, 5 = Very high

Cleanliness level – 1 = Acceptable, 2 = Poor, 3 = Very Poor, 4 = Good, 5 = Excellent

Energy Consumption – 1 = Very Low, 2 = Low, 3 = Medium, 4 = High, 5 = Very High.

New Contamination Thickness – 1 = Low, 2 = Medium, 3 = High, 0 = None

Weights all set to (1). Concatenation = 1 (AND).

```
[System]
Name='Fuzzy_Memberships_and_Rules_New_2'
Type='mamdani'
Version=2.0
NumInputs=6
NumOutputs=4
NumRules=193
AndMethod='prod'
OrMethod='max'
ImpMethod='prod'
AggMethod='max'
DefuzzMethod='centroid'

[Input1]
Name='Contaminant_Type'
Range=[0 3.1]
NumMFs=3
MF1='Organic':'trimf',[0.99 1 1.01]
MF2='Inorganic':'trimf',[2.056 2.067 2.077]
MF3='Mixed':'trimf',[2.99 3 3.01]

[Input2]
Name='Contamination_Thickness'
Range=[0 4]
NumMFs=3
MF1='Low':'trimf',[-1 0 1.5]
MF2='Medium':'trimf',[0.5 2 3.5]
MF3='High':'trimf',[2.5 4 5]

[Input3]
Name='Material_Type'
Range=[0 2]
NumMFs=2
MF1='Metal':'trimf',[0.99 1 1.01]
MF2='Non-Metal':'trimf',[1.99 2 2.01]

[Input4]
Name='Product_Shape'
Range=[0 3]
NumMFs=3
MF1='Simple':'trimf',[-0.1 0 1.2]
MF2='Complex':'trimf',[0.5 1.5 2.5]
MF3='Very_Complex':'trimf',[1.8 3 4.2]

[Input5]
Name='Cleaning_Process'
Range=[0 7]
NumMFs=8
MF1='Non_Process':'trimf',[-2.8 0 0.1]
MF2='Laser':'trimf',[2.9 3 3.1]
MF3='Thermal':'trimf',[3.88148148148148 3.98148148148148 4.08148148148148]
MF4='Ultrasonic':'trimf',[0.9 1 1.1]
MF5='Abrasive':'trimf',[1.9 2 2.1]
MF6='Immersion':'trimf',[4.9 5 5.1]
```

```
MF7='Molten_Salt':'trimf',[5.88148148148148 5.98148148148148
6.08148148148148]
MF8='Vibratory':'trimf',[6.9 7 7.1]

[Input6]
Name='Cleaning_Process_Time'
Range=[0 60]
NumMFs=5
MF1='Very_Fast':'trapmf',[-1 -1 5 10]
MF2='Fast':'trimf',[7 13.5 20]
MF3='Medium':'trimf',[15 23 30]
MF4='Slow':'trimf',[26 35 45]
MF5='Very_Slow':'trapmf',[40 50 60 60]

[Output1]
Name='Cost'
Range=[0 450000]
NumMFs=5
MF1='Very_Low':'trapmf',[0 0 40000 100000]
MF2='Low':'trimf',[60000 125000 195000]
MF3='Medium':'trimf',[135000 210000 285000]
MF4='High':'trimf',[235000 295000 355000]
MF5='Very_High':'trapmf',[300000 400000 450000 460000]

[Output2]
Name='Cleaning_Level'
Range=[0 10]
NumMFs=5
MF1='Acceptable':'trimf',[3.78901734104046 5.28901734104046 6.78901734104046]
MF2='Poor':'trimf',[1.65028901734104 3.15028901734104 4.65028901734104]
MF3='Good':'trimf',[5.84682080924856 7.34682080924856 8.84682080924856]
MF4='Very_Poor':'trapmf',[-9 -1 1 2.5]
MF5='Excellent':'trapmf',[8.23988439306358 10.2398843930636 10.7398843930636
11.7398843930636]

[Output3]
Name='Energy_Consumption'
Range=[0 10]
NumMFs=5
MF1='Very_Low':'trapmf',[-5.43 0 0.5 2]
MF2='Low':'trimf',[1 2.5 4]
MF3='Medium':'trimf',[3 4.5 6]
MF4='High':'trimf',[5 6.5 8]
MF5='Very_high':'trapmf',[7 9 10 11.7]

[Output4]
Name='New_Contamination_Thickness'
Range=[0 4]
NumMFs=3
MF1='Low':'trimf',[-1 0 1.5]
MF2='Medium':'trimf',[0.5 2 3.5]
MF3='High':'trimf',[2.5 4 5]

[Rules]
0 1 0 0 1 0, 0 0 0 1 (1) : 1
0 2 0 0 1 0, 0 0 0 2 (1) : 1
```

```
0 3 0 0 1 0, 0 0 0 3 (1) : 1
-1 -3 1 1 2 -1, 2 5 3 1 (1) : 1
-1 -3 1 1 2 1, 2 5 5 1 (1) : 1
-1 -3 1 2 2 1, 3 3 5 1 (1) : 1
-1 -3 1 3 2 -1, 4 1 5 1 (1) : 1
-1 -3 1 3 2 1, 4 1 5 1 (1) : 1
1 -3 1 1 2 -1, 2 5 5 1 (1) : 1
1 -3 1 1 2 1, 2 5 3 1 (1) : 1
1 -3 1 2 2 1, 3 3 3 1 (1) : 1
1 -3 1 3 2 -1, 4 1 3 1 (1) : 1
1 -3 1 3 2 1, 4 1 3 1 (1) : 1
-1 3 1 1 2 -1, 2 5 3 1 (1) : 1
-1 3 1 1 2 1, 2 3 3 1 (1) : 1
-1 3 1 2 2 -1, 3 3 3 1 (1) : 1
-1 3 1 2 2 1, 3 3 3 1 (1) : 1
-1 3 1 3 2 1, 4 2 3 1 (1) : 1
1 3 1 1 2 -1, 2 5 3 1 (1) : 1
1 3 1 1 2 1, 2 3 3 1 (1) : 1
1 3 1 2 2 -1, 3 3 3 1 (1) : 1
1 3 1 2 2 1, 3 3 3 1 (1) : 1
1 3 1 3 2 1, 4 2 3 1 (1) : 1
2 1 0 0 3 0, 5 4 5 1 (1) : 1
1 1 1 0 3 1, 1 2 1 1 (1) : 1
1 1 1 0 3 2, 1 1 1 1 (1) : 1
1 1 1 0 3 3, 1 3 2 1 (1) : 1
1 1 1 0 3 4, 1 3 2 1 (1) : 1
1 1 1 0 3 5, 1 5 3 1 (1) : 1
1 2 1 0 3 1, 1 4 1 2 (1) : 1
1 2 1 0 3 2, 1 2 1 2 (1) : 1
1 2 1 0 3 3, 1 1 2 1 (1) : 1
1 2 1 0 3 4, 1 1 2 1 (1) : 1
1 2 1 0 3 5, 1 3 3 1 (1) : 1
1 3 1 0 3 1, 1 4 1 3 (1) : 1
1 3 1 0 3 2, 1 4 1 3 (1) : 1
1 3 1 0 3 3, 1 3 2 2 (1) : 1
1 3 1 0 3 4, 1 2 2 2 (1) : 1
1 3 1 0 3 5, 1 1 3 2 (1) : 1
3 1 1 0 3 1, 1 4 1 1 (1) : 1
3 1 1 0 3 2, 1 4 1 1 (1) : 1
3 1 1 0 3 3, 1 2 2 1 (1) : 1
3 1 1 0 3 4, 1 2 2 1 (1) : 1
3 1 1 0 3 5, 1 1 3 1 (1) : 1
3 2 1 0 3 1, 1 4 1 2 (1) : 1
3 2 1 0 3 2, 1 4 1 2 (1) : 1
3 2 1 0 3 3, 1 2 2 2 (1) : 1
3 2 1 0 3 4, 1 2 2 2 (1) : 1
3 2 1 0 3 5, 1 1 3 1 (1) : 1
3 3 1 0 3 1, 1 4 1 3 (1) : 1
3 3 1 0 3 2, 1 4 1 3 (1) : 1
3 3 1 0 3 3, 1 2 2 3 (1) : 1
3 3 1 0 3 4, 1 2 2 3 (1) : 1
3 3 1 0 3 4, 1 1 3 2 (1) : 1
1 1 0 0 4 0, 5 4 5 1 (1) : 1
2 1 0 0 4 1, 1 1 1 1 (1) : 1
2 1 0 0 4 2, 1 3 2 1 (1) : 1
2 1 0 0 4 3, 2 3 3 1 (1) : 1
2 1 0 0 4 4, 2 5 4 1 (1) : 1
```

```
2 1 0 0 4 5, 2 5 5 1 (1) : 1
2 2 0 0 4 1, 1 2 1 2 (1) : 1
2 2 0 0 4 2, 2 1 2 1 (1) : 1
2 2 0 0 4 3, 2 3 3 1 (1) : 1
2 2 0 0 4 4, 2 5 4 1 (1) : 1
2 2 0 0 4 5, 2 5 5 1 (1) : 1
2 3 0 0 4 1, 1 4 1 3 (1) : 1
2 3 0 0 4 2, 1 3 2 3 (1) : 1
2 3 0 0 4 3, 2 2 3 2 (1) : 1
2 3 0 0 4 4, 2 1 4 2 (1) : 1
2 3 0 0 4 5, 2 1 5 1 (1) : 1
3 1 0 0 4 1, 1 4 1 1 (1) : 1
3 1 0 0 4 2, 1 2 2 1 (1) : 1
3 1 0 0 4 3, 2 1 3 1 (1) : 1
3 1 0 0 4 4, 2 1 4 1 (1) : 1
3 1 0 0 4 5, 2 3 5 1 (1) : 1
3 2 0 0 4 1, 1 4 1 2 (1) : 1
3 2 0 0 4 2, 1 4 2 2 (1) : 1
3 2 0 0 4 3, 2 2 3 2 (1) : 1
3 2 0 0 4 4, 2 1 4 1 (1) : 1
3 2 0 0 4 5, 2 1 5 1 (1) : 1
3 3 0 0 4 1, 1 4 1 3 (1) : 1
3 3 0 0 4 2, 1 4 2 3 (1) : 1
3 3 0 0 4 3, 2 2 3 3 (1) : 1
3 3 0 0 4 4, 2 2 4 2 (1) : 1
3 3 0 0 4 5, 2 1 5 2 (1) : 1
0 1 0 -3 5 1, 1 3 2 1 (1) : 1
0 1 0 -3 5 2, 1 5 3 1 (1) : 1
0 1 0 -3 5 3, 1 5 4 1 (1) : 1
0 1 0 -3 5 4, 2 5 5 1 (1) : 1
0 1 0 -3 5 5, 2 5 5 1 (1) : 1
0 2 0 -3 5 1, 1 1 2 1 (1) : 1
0 2 0 -3 5 2, 1 1 3 1 (1) : 1
0 2 0 -3 5 3, 1 3 4 1 (1) : 1
0 2 0 -3 5 4, 2 3 5 1 (1) : 1
0 2 0 -3 5 5, 2 5 5 1 (1) : 1
0 3 0 -3 5 1, 1 2 2 3 (1) : 1
0 3 0 -3 5 2, 1 1 3 2 (1) : 1
0 3 0 -3 5 3, 2 1 4 2 (1) : 1
0 3 0 -3 5 4, 2 3 5 1 (1) : 1
0 3 0 -3 5 5, 2 3 5 1 (1) : 1
0 1 0 3 5 1, 1 4 2 1 (1) : 1
0 1 0 3 5 2, 1 4 3 1 (1) : 1
0 1 0 3 5 3, 1 2 4 1 (1) : 1
0 1 0 3 5 4, 2 1 5 1 (1) : 1
0 1 0 3 5 5, 2 1 5 1 (1) : 1
0 2 0 3 5 1, 1 4 1 2 (1) : 1
0 2 0 3 5 2, 1 4 2 2 (1) : 1
0 2 0 3 5 3, 2 2 3 1 (1) : 1
0 2 0 3 5 4, 2 1 4 1 (1) : 1
0 2 0 3 5 5, 2 1 5 1 (1) : 1
0 3 0 3 5 1, 1 4 2 3 (1) : 1
0 3 0 3 5 2, 1 4 3 3 (1) : 1
0 3 0 3 5 3, 1 2 4 3 (1) : 1
0 3 0 3 5 4, 2 2 5 2 (1) : 1
0 3 0 3 5 5, 2 2 5 2 (1) : 1
2 1 0 0 6 0, 5 4 5 1 (1) : 1
```

```
1 1 0 0 6 1, 1 5 2 1 (1) : 1
1 1 0 0 6 2, 1 5 3 1 (1) : 1
1 1 0 0 6 3, 1 5 4 1 (1) : 1
1 1 0 0 6 4, 1 5 5 1 (1) : 1
1 1 0 0 6 5, 1 5 5 1 (1) : 1
1 2 0 0 6 1, 1 3 2 2 (1) : 1
1 2 0 0 6 2, 1 5 3 1 (1) : 1
1 2 0 0 6 3, 1 5 4 1 (1) : 1
1 2 0 0 6 4, 1 5 5 1 (1) : 1
1 2 0 0 6 5, 1 5 5 1 (1) : 1
1 3 0 0 6 1, 1 1 2 3 (1) : 1
1 3 0 0 6 2, 1 3 3 2 (1) : 1
1 3 0 0 6 3, 1 5 4 1 (1) : 1
1 3 0 0 6 4, 1 5 5 1 (1) : 1
1 3 0 0 6 5, 1 5 5 1 (1) : 1
3 1 0 0 6 1, 1 2 2 1 (1) : 1
3 1 0 0 6 2, 1 2 3 1 (1) : 1
3 1 0 0 6 3, 1 1 4 1 (1) : 1
3 1 0 0 6 4, 1 1 5 1 (1) : 1
3 1 0 0 6 5, 1 1 5 1 (1) : 1
3 2 0 0 6 1, 1 4 2 2 (1) : 1
3 2 0 0 6 2, 1 2 3 1 (1) : 1
3 2 0 0 6 3, 1 2 4 1 (1) : 1
3 2 0 0 6 4, 1 1 5 1 (1) : 1
3 2 0 0 6 5, 1 1 5 1 (1) : 1
3 3 0 0 6 1, 1 4 2 3 (1) : 1
3 3 0 0 6 2, 1 4 3 2 (1) : 1
3 3 0 0 6 3, 1 2 4 2 (1) : 1
3 3 0 0 6 4, 1 2 5 1 (1) : 1
3 3 0 0 6 5, 1 1 5 1 (1) : 1
0 1 1 0 7 1, 1 1 1 1 (1) : 1
0 1 1 0 7 2, 1 3 2 1 (1) : 1
0 1 1 0 7 3, 1 5 3 1 (1) : 1
0 1 1 0 7 4, 1 5 4 1 (1) : 1
0 1 1 0 7 5, 1 5 5 1 (1) : 1
0 2 1 0 7 1, 1 2 1 2 (1) : 1
0 2 1 0 7 2, 1 2 2 1 (1) : 1
0 2 1 0 7 3, 1 1 3 1 (1) : 1
0 2 1 0 7 4, 1 3 4 1 (1) : 1
0 2 1 0 7 5, 1 5 5 1 (1) : 1
0 3 1 0 7 1, 1 4 1 3 (1) : 1
0 3 1 0 7 2, 1 2 2 3 (1) : 1
0 3 1 0 7 3, 1 1 3 2 (1) : 1
0 3 1 0 7 4, 1 3 4 1 (1) : 1
0 3 1 0 7 5, 1 3 5 1 (1) : 1
2 2 0 0 3 0, 5 4 5 2 (1) : 1
2 3 0 0 3 0, 5 4 5 3 (1) : 1
1 2 0 0 4 0, 5 4 5 2 (1) : 1
1 3 0 0 4 0, 5 4 5 3 (1) : 1
2 2 0 0 6 0, 5 4 5 2 (1) : 1
2 3 0 0 6 0, 5 4 5 3 (1) : 1
0 1 1 -3 8 1, 1 3 1 1 (1) : 1
0 2 1 -3 8 5, 1 5 3 1 (1) : 1
0 2 1 -3 8 4, 1 5 2 1 (1) : 1
0 2 1 -3 8 3, 1 3 2 1 (1) : 1
0 2 1 -3 8 2, 1 3 1 1 (1) : 1
0 2 1 -3 8 1, 1 3 1 1 (1) : 1
```

```
0 3 1 -3 8 5, 1 3 3 1 (1) : 1
0 3 1 -3 8 4, 1 3 2 1 (1) : 1
0 3 1 -3 8 3, 1 1 2 2 (1) : 1
0 3 1 -3 8 2, 1 2 1 2 (1) : 1
0 3 1 -3 8 1, 1 4 1 3 (1) : 1
0 1 1 3 8 5, 1 1 3 1 (1) : 1
0 1 1 3 8 4, 1 1 2 1 (1) : 1
0 1 1 3 8 3, 1 1 2 1 (1) : 1
0 1 1 3 8 2, 1 2 1 1 (1) : 1
0 1 1 3 8 1, 1 2 1 1 (1) : 1
0 2 1 3 8 5, 1 1 3 1 (1) : 1
0 2 1 3 8 4, 1 1 2 1 (1) : 1
0 2 1 3 8 3, 1 1 2 1 (1) : 1
0 2 1 3 8 2, 1 2 1 2 (1) : 1
0 2 1 3 8 1, 1 2 1 2 (1) : 1
0 3 1 3 8 5, 1 1 3 2 (1) : 1
0 3 1 3 8 4, 1 1 2 2 (1) : 1
0 3 1 3 8 3, 1 2 2 2 (1) : 1
0 3 1 3 8 1, 1 4 1 3 (1) : 1
0 3 1 3 8 2, 1 4 1 3 (1) : 1
```

APPENDIX C

APPENDIX C

MATLAB GRAPHICAL USER INTERFACE CODE

```
function varargout = Cleaning_Processes(varargin)
% CLEANING_PROCESSES M-file for Cleaning_Processes.fig
%      CLEANING_PROCESSES, by itself, creates a new CLEANING_PROCESSES or raises
the existing
%      singleton*.
%
%      H = CLEANING_PROCESSES returns the handle to a new CLEANING_PROCESSES or
the handle to
%      the existing singleton*.
%
%      CLEANING_PROCESSES('CALLBACK',hObject,eventData,handles,...) calls the
local
%      function named CALLBACK in CLEANING_PROCESSES.M with the given input
arguments.
%
%      CLEANING_PROCESSES('Property','Value',...) creates a new
CLEANING_PROCESSES or raises the
%      existing singleton*.  Starting from the left, property value pairs are
%      applied to the GUI before Cleaning_Processes_OpeningFcn gets called.  An
%      unrecognized property name or invalid value makes property application
%      stop.  All inputs are passed to Cleaning_Processes_OpeningFcn via
varargin.
%
%      *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows only one
%      instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help Cleaning_Processes

% Last Modified by GUIDE v2.5 27-May-2014 17:15:38

% Begin initialization code - DO NOT EDIT
```

```
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @Cleaning_Processes_OpeningFcn, ...
                   'gui_OutputFcn',  @Cleaning_Processes_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);

if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT


% --- Executes just before Cleaning_Processes is made visible.
function Cleaning_Processes_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to Cleaning_Processes (see VARARGIN)

% Choose default command line output for Cleaning_Processes
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes Cleaning_Processes wait for user response (see UIRESUME)
% uiwait(handles.figure1);


% --- Outputs from this function are returned to the command line.
function varargout = Cleaning_Processes_OutputFcn(hObject, eventdata, handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;


% --- Executes on slider movement.
% --- Contamination Thickness Level
function slider1_Callback(hObject, eventdata, handles)
% hObject    handle to slider1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global CTh;
CTh = get(hObject, 'Value');
set(handles.pushbutton1,'String',CTh);
% Hints: get(hObject,'Value') returns position of slider
%        get(hObject,'Min') and get(hObject,'Max') to determine range of slider


% --- Executes during object creation, after setting all properties.
function slider1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to slider1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: slider controls usually have a light gray background.
if                                       isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end


% --- Executes on slider movement.
function slider2_Callback(hObject, eventdata, handles)
% hObject    handle to slider2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global PS;
PS = get(hObject, 'Value');
set(handles.pushbutton2,'String',PS);

% Hints: get(hObject,'Value') returns position of slider
%        get(hObject,'Min') and get(hObject,'Max') to determine range of slider


% --- Executes during object creation, after setting all properties.
function slider2_CreateFcn(hObject, eventdata, handles)
% hObject    handle to slider2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: slider controls usually have a light gray background.
if                                       isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end
```

```
% --- Executes on button press in radiobutton2.
function radiobutton2_Callback(hObject, eventdata, handles)
% hObject    handle to radiobutton2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
set(handles.radiobutton4,'Value',0);
set(handles.radiobutton5,'Value',0);
% Hint: get(hObject,'Value') returns toggle state of radiobutton2
% --- Executes on button press in radiobutton4.
function radiobutton4_Callback(hObject, eventdata, handles)
% hObject    handle to radiobutton4 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of radiobutton4


% --- Executes on button press in radiobutton5.
function radiobutton5_Callback(hObject, eventdata, handles)
% hObject    handle to radiobutton5 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of radiobutton5


% --- Executes on button press in radiobutton7.
function radiobutton7_Callback(hObject, eventdata, handles)
% hObject    handle to radiobutton7 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of radiobutton7


% --- Executes on button press in radiobutton8.
function radiobutton8_Callback(hObject, eventdata, handles)
% hObject    handle to radiobutton8 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of radiobutton8



function edit2_Callback(hObject, eventdata, handles)
% hObject    handle to edit2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% Hints: get(hObject,'String') returns contents of edit2 as text
%        str2double(get(hObject,'String')) returns contents of edit2 as a double


% --- Executes during object creation, after setting all properties.
function edit2_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if          ispc          &&          isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on button press in pushbutton1.
function pushbutton1_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
set(hObject,'BackgroundColor',[.9 .9 .9]);


% --- Executes on button press in pushbutton2.
function pushbutton2_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
set(hObject,'BackgroundColor',[.9 .9 .9]);


% --- Executes on button press in pushbutton3.
function pushbutton3_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global Q
global CT
global CTh
global MT
global PS
global Tw
global CLp
global Ev
Q = Thesis_Program_Cleaning_Processes(CT,CTh,MT,PS,Tw,CLp,Ev)
```

```
% --- Executes on slider movement.
function slider6_Callback(hObject, eventdata, handles)
% hObject    handle to slider6 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global Tw
Tw = get(hObject, 'Value');
set(handles.pushbutton4,'String',Tw);


% Hints: get(hObject,'Value') returns position of slider
%        get(hObject,'Min') and get(hObject,'Max') to determine range of slider


% --- Executes during object creation, after setting all properties.
function slider6_CreateFcn(hObject, eventdata, handles)
% hObject    handle to slider6 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: slider controls usually have a light gray background.
if                                       isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end


% --- Executes on button press in pushbutton4.
function pushbutton4_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton4 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
set(hObject,'BackgroundColor',[.9 .9 .9]);


% --- Executes on slider movement.
function slider7_Callback(hObject, eventdata, handles)
% hObject    handle to slider7 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global Ev
Ev = get(hObject, 'Value');
set(handles.pushbutton5,'String',Ev);

% Hints: get(hObject,'Value') returns position of slider
%        get(hObject,'Min') and get(hObject,'Max') to determine range of slider


% --- Executes during object creation, after setting all properties.
function slider7_CreateFcn(hObject, eventdata, handles)
```

```
% hObject    handle to slider7 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: slider controls usually have a light gray background.
if                                     isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end



% --- Executes on button press in pushbutton5.
function pushbutton5_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton5 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
set(hObject,'BackgroundColor',[.9 .9 .9]);



% --- Executes on slider movement.
function slider8_Callback(hObject, eventdata, handles)
% hObject    handle to slider8 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global CLp
CLp = get(hObject, 'Value');
set(handles.pushbutton6,'String',CLp);

% Hints: get(hObject,'Value') returns position of slider
%        get(hObject,'Min') and get(hObject,'Max') to determine range of slider



% --- Executes during object creation, after setting all properties.
function slider8_CreateFcn(hObject, eventdata, handles)
% hObject    handle to slider8 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: slider controls usually have a light gray background.
if                                     isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end



% --- Executes on button press in pushbutton6.
function pushbutton6_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton6 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
```

```
set(hObject,'BackgroundColor',[.9 .9 .9])


% --- Executes on selection change in popupmenu2.
function popupmenu2_Callback(hObject, eventdata, handles)
% hObject    handle to popupmenu2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns popupmenu2 contents
as cell array
%        contents{get(hObject,'Value')} returns selected item from popupmenu2
global CT
CT = get(hObject,'Value');

% --- Executes during object creation, after setting all properties.
function popupmenu2_CreateFcn(hObject, eventdata, handles)
% hObject    handle to popupmenu2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: popupmenu controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if          ispc          &&          isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
set(hObject,'String',{'Organic';'Inorganic';'Mixed'})


% --- Executes on selection change in popupmenu3.
function popupmenu3_Callback(hObject, eventdata, handles)
% hObject    handle to popupmenu3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns popupmenu3 contents
as cell array
%        contents{get(hObject,'Value')} returns selected item from popupmenu3
global MT
MT = get(hObject,'Value');
% temp1 = get(hObject,'string');
% temp2 = get(hObject,'Value');
% MT = temp1(temp2);
% switch temp2
%     case 1
%         MT = 1
%     case 2
%         MT = 2
% end
% --- Executes during object creation, after setting all properties.
function popupmenu3_CreateFcn(hObject, eventdata, handles)
% hObject    handle to popupmenu3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: popupmenu controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if          ispc          &&          isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
set(hObject,'String',{'Metal';'Non-Metal'})

function edit3_Callback(hObject, eventdata, handles)
% hObject    handle to edit3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit3 as text
%       str2double(get(hObject,'String')) returns contents of edit3 as a double


% --- Executes during object creation, after setting all properties.
function edit3_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if          ispc          &&          isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit4_Callback(hObject, eventdata, handles)
% hObject    handle to edit4 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit4 as text
%       str2double(get(hObject,'String')) returns contents of edit4 as a double


% --- Executes during object creation, after setting all properties.
function edit4_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit4 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called
% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if          ispc          &&          isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

BIOGRAPHICAL SKETCH

Juan Carlos Martinez Chacin was born in Maracaibo, Venezuela on October 25th, 1987. At the age of sixteen he graduated from High School at started his professional career at the "Universidad Simón Bolívar" majoring in Electrical Engineering. As time passed and knowledge gathered, he decided to change his major to Production Engineering because it best suited his personality. Due to problems arising in his home country, he moved to Monterrey, México. In 2008, he continued his studies at the "Universidad Regiomontana" now with the major of Industrial and Systems Engineering (similar the one he studied in Venezuela). During his study in Mexico, he worked as Intern in Ternium Mexico, S.A de C.V. as a training coordinator of the Continuous Improvement course. He got the skills of negotiation, course logistics and data gathering. By the time he graduated in 2010 and receiving his Bachelor's degree, he started working in a company called Detailing Multemedia S.A. de C.V as Project & Sales Manager. He improved the inventory control, the customer and supplier relationship, response time for problem fixing and order fulfillment by applying Lean Manufacturing and Six Sigma knowledge.

In August 2011 he decided to pursue a Masters in Engineering Management – Systems concentration thanks to the recommendation made by the program director Dr. Alley Butler. Here, he met excellent professors such as Dr. Jianzhi Li, Dr. Miguel Gonzalez, Dr. Douglar Timmer and Dr. Hiram Moya. Also, he worked as Teacher Assistant at the Civil Engineer Department. He conducted a research in Remanufacturing, specifically in the cleaning processes for process planning. For any contact, please write to martinezchacin.juancarlos@gmail.com