Development of a time-efficient heuristic method for production scheduling with resource constraints and changeover considerations

by

Jennifer Muñoz Blás

A thesis submitted in partial fulfillment of the requirements for the degree of
MASTER IN SCIENCE
in

INDUSTRIAL ENGINEERING
UNIVERSITY OF PUERTO RICO
MAYAGŰEZ CAMPUS

December, 2007

Approved by:

________________________	________
Pedro Resto, Ph.D. Chairperson, Graduate Committee	Date
________________________	________
Mario Padrón, Ph.D. Member, Graduate Committee	Date
________________________	________
Sonia Bartolomei, Ph.D. Member, Graduate Committee	Date
________________________	________
Viviana Cesaní, Ph.D. Member, Graduate Committee	Date
________________________	________
Agustín Rullán, Ph.D. Department Chairperson	Date
________________________	________
José R. Cedeño Graduate School Representative	Date

RESUMEN

Usar la herramienta correcta en la planificación de los recursos de producción es absolutamente necesario para que las industrias puedan competir eficazmente y tener éxito en la actual economía global. El método heurístico desarrollado en esta investigación es uno de tiempo computacional eficiente que permite resolver problemas de planificación de producción con restricciones de recursos considerando el tiempo de ajuste por productos o partes. La efectividad de los resultados obtenidos por el método heurístico fue comparada usando un modelo de programación entera mixta desarrollado por Stefan Voss. Este modelo toma en consideración los tiempos de ajuste de máquinas y la capacidad disponible al momento de planificar la producción.

Tanto el modelo de optimización como el método heurístico fueron programados y ejecutados en el lenguaje “Visual Basic”. Se diseñó y ejecutó un experimento estadístico para determinar cuán cercana estaba la solución del método heurístico del valor óptimo y cuánto tiempo se economiza haciendo uso del método heurístico. Teniendo esta información analizada, se procedió a desarrollar un caso de estudio que provee la complejidad necesaria para evaluar ambos modelos con un diseño de experimento. Usando análisis de regresión se evaluó la diferencia entre la función objetivo y el tiempo computacional de los métodos y se demostró que el método heurístico provee una solución bien cercana al óptimo haciendo uso siempre de un tiempo computacional mínimo.

ABSTRACT

The use of the correct tool in the resource production planning process is absolutely necessary in order for industries to compete effectively and be successful in the actual global economy. The heuristic method developed as part of this research is computationally time efficient and allows solving production planning problems with restrictions of resources and considering product or parts changeover times. The proximity of the results obtained by the heuristic method was compared against a mix integer linear programming model developed by Stefan Voss. This model consideres the setup times and the capacity available for production.

Both the optimization model and the heuristic method where programmed and executed in “Visual BASIC”. A statistical experiment was designed and executed to determine how close the solution of the heuristic method was from the optimal value and the computational time savings when the heuristic method is used. Having this information analyzed, a case study that provides the necessary complexity to evaluate both models was developed using an experimental design. Using regression analysis and normality tests for the residuals, the difference between objective function value and computational time from both methods demonstrates that the heuristic method provides a solution near the optimum value using always minimal computational time.

I specially dedicate this work to my beloved husband Héctor Edwin who gave me all the support and love to overcome the pressure that sometimes I felt. To my parents Rosario and Rafael, and my friends Alexis and Anne that help me find the way back every time I felt lost. To my mentors, Mario and Pedro who guided me and taught me the true value of teamwork.

ACKNOWLEDGEMENT

I am thankful mainly to God for all the good people that have crossed my way and that helped me on the development of this thesis. I thank my advisor the Dr. Pedro Resto for giving me the opportunity to develop this investigation under his guidance and for his kindness and professionalism.

I am very grateful to Dr Mario Padrón who gave me the idea for this research. Dr. Padrón was also my lead and advisor on the programming process, transmitting his knowledge on visual basic language and also for teaching me what takes to be successful.

My deepest thanks to Dr. Pedro Resto for all his good ideas, enthusiasm, optimism, charisma and wisdom that help me move forward. Thanks to my professors in the graduate program: Dra. Sonia Bartolomei, Dra. Viviana Cesaní, and Dr. David Gonzalez. Special thanks to the secretaries of the Industrial Engineering department, Mayra Colón and Laura González for their kindness and support. I am also thankful to Edwin Morales and the professors that contributed during the experimentation process; thanks for all your understanding and help. My deepest appreciation to all my good peers in the masters program, especially to Heidy and Paola.

TABLE OF CONTENT

CHAPTER 1: INTRODUCTION **1**

1.1 JUSTIFICATION 6

1.2 CONTRIBUTION 7

1.3 OBJECTIVES 7

CHAPTER 2: LITERATURE REVIEW **9**

2.1. PLANNING TOOLS EVOLUTION 9

2.2. MATERIAL REQUIREMENT PLANNING 10

2.3. MATERIAL RESOURCES PLANNING 11

2.3.1. Enterprise Resource Planning *14*

2.3.2. Supply Chain Management *15*

2.4. OPTIMIZATION MODELS IN PRODUCTION PLANNING 16

2.4.1. Advanced Planning and Optimizing *21*

2.4.2. New approaches to optimization *23*

2.5. HEURISTIC MODELS IN PRODUCTION PLANNING 24

CHAPTER 3: METHODOLOGY **31**

3.1 DETAILED DESCRIPTION OF METHODOLOGY STEPS 32

3.2 CASE STUDY DEVELOPED AS PART OF THIS RESEARCH 36

3.3 COST COMPONENTS FOR THE OPTIMIZATION AND HEURISTIC MODELS 40

3.4 DETAILED DESCRIPTION OF THE CONSTRAINTS USED IN THE MODELS 40

CHAPTER 4: RESULTS **41**

4.1 DESIGN OF THE EXPERIMENT TO COMPARE THE ALGORITHM AND THE HEURISTIC APPROACHES 41

4.2 RESULTS OBTAINED FROM THE EXPERIMENT 44

4.2.1 Analysis of Results Considering Blocking 44

4.2.2 Analysis of Results Considering Differences 48

4.2.4 Analysis of Results for Computational Time Requirements 53

CHAPTER 5: CONCLUSIONS AND FUTURE RESEARCH 56

BIBLIOGRAPHY 58

APPENDIX A. VISUAL BASIC CODE FOR HEURISTIC MODEL 62

APPENDIX B. BEHAVIOR OF THE HEURISTIC APPROACH 87

EXPERIMENT #1 87

EXPERIMENT #2 88

EXPERIMENT #3 89

EXPERIMENT #4 90

EXPERIMENT #5 91

EXPERIMENT #6 92

EXPERIMENT #7 93

EXPERIMENT #8 94

EXPERIMENT #9 95

EXPERIMENT #10 96

EXPERIMENT #11 97

EXPERIMENT #12 98

EXPERIMENT #13 99

EXPERIMENT #14 100

EXPERIMENT #15 101

EXPERIMENT #16 102

LIST OF FIGURES

FIGURE 1 MRP II MODEL 2

FIGURE 2: METHODOLOGY FLOWCHART 31

FIGURE 3: HEURISTIC LOGIC FLOWCHART 34

FIGURE 4: PRODUCTION LINE DIAGRAM 37

FIGURE 5: EXAMPLE OF WORKBOOK USE TO RUN THE OPTIMIZATION MODEL AND THE HEURISTIC MODEL 37

FIGURE 6: REGULAR FLASHLIGHT DIAGRAM 38

FIGURE 7: EMERGENCY FLASHLIGHT DIAGRAM 39

FIGURE 8: REGRESSION ANALYSIS FOR THE COMPLETE MODEL 45

FIGURE 9: RESIDUAL ANALYSIS FOR FULL MODEL 45

FIGURE 10: REGRESSION ANALYSIS FOR THE SIMPLIFIED MODEL 46

FIGURE 11: RESIDUAL ANALYSIS FOR SIMPLIFIED MODEL 46

FIGURE 12: OBJECTIVE FUNCTION VALUES 47

FIGURE 13: KOLMOGOROV-SMIRNOV TEST FOR NORMALITY ON THE SIMPLIFIED MODEL 47

FIGURE 14: REGRESSION ANALYSIS FOR DIFFERENCE 49

FIGURE 15: RESIDUAL ANALYSIS FOR DIFFERENCES 49

FIGURE 16: NORMAL PROBABILITY PLOT FOR RESIDUALS OF THE DIFFERENCE MODEL 50

FIGURE 17: COMPARISON BETWEEN OBJECTIVE FUNCTION FOR ALGORITHM & HEURISTIC MODEL 51

FIGURE 18: COMPARISON BETWEEN OBJECTIVE FUNCTION FROM ALGORITHM & HEURISTIC 51

FIGURE 19: MAIN EFFECTS FOR THE DIFFERENCE MODEL 52

FIGURE 20: TWO-FACTOR INTERACTIONS FOR THE DIFFERENCE MODEL 52

FIGURE 21: REGRESSION ANALYSIS FOR DIFFERENCE IN COMPUTATIONAL TIME 53

FIGURE 22: RESIDUAL PLOT FOR DIFFERENCE IN TIME REQUIREMENTS 54

FIGURE 23: BOXPLOT OF COMPUTATIONAL TIME REQUIREMENTS 55

LIST OF TABLES

TABLE 1: BENEFITS ACHIEVED FROM MRP Π IMPLEMENTATION.................................. 3

TABLE 2: DATA FOR A MRP Π FORMULATION.. 4

TABLE 3: TASK-TECHNOLOGY INTEGRATION IN MRP, MRPΠ AND ERP...................... 14

TABLE 4: PARAMETER AND DECISION VARIABLES FOR THE GLSP MODEL............... 22

TABLE 5: HEURISTIC'S FIRST THREE STEPS 25

TABLE 6: INITIAL DATA REQUIRE BY MODEL 35

TABLE 7: REGULAR FLASHLIGHT BOM... 38

TABLE 8: EMERGENCY FLASHLIGHT BOM 39

TABLE 9: FACTORS AND LEVELS DEFINED IN THE EXPERIMENTAL 42

TABLE 10: OBJECTIVE FUNCTION RESULT BY EXPERIMENTAL CONDITION 43

Chapter 1: Introduction

Material requirements planning (MRP) systems have become the most effective and widely used inventory control systems across the world [17]. Many operation managers have found the great value of MRP systems as a production planning tool. Such tool is absolutely necessary to effectively and competitively succeed in the current global economy. MRP calculates material needs by computing raw material and subassembly needs to comply with customer demand for a specific point in time.

MRP originated in the early 1970s in the USA as a computerized approach for the planning of materials acquisition and production. The technique had undoubtedly been manually practiced in aggregate form prior to the World War Π in several locations in Europe. These early-computerized applications of MRP were built around a bill of material processor (BOMP), which converted a discrete plan of production for a parent item into a discrete plan of production or purchasing for component items.

With the introduction of MRP, three main advantages came about. The first advantage involved statistical forecasting for components with “lumpy” demand. A second benefit was that MRP systems provide managers with more information than was possible under older inventory control systems. Finally, MRP systems updated dependent demand and replenishment schedules of components as the schedules of the parent items change [21]. MRP systems alerted planners when a change in production levels (up or down) is needed.

Over the last two decades, researchers and practitioners have developed and implemented many optimal planning models and methods for material purchasing, shopfloor scheduling, capacity planning and other production functions. But there are a number of well-known and very severe problems with MRP optimization models [25]. The three most serious problems are :

- The actual time to complete an order is usually a function of congestion rather than the SKU.
- Lot sizing can cause nervousness.
- There are no capacity constraints, capacity is presumed infinite.

Potentially the most severe is the fact that it ignores capacity; this weakens its usefulness when used as a scheduling tool, but is also bad for MRP as a planning tool. The absence of capacity considerations can make the plans so unrealistic that they are not useful. For these and others reasons, as the ever increasing competitiveness of the world market, companies have been forced to optimize their operations to win business and develop optimal models for manufacturing resource planning (MRP Π). The manufacturing resources planning system, MRP Π is shown in Figure 1. MRP Π is designed to link manufacturing with other functions, such as engineering, marketing, finance, and human resources. The more popular MRP Π software available includes SSA, ForthShift, QAD, EMS, AvaLon, SAP, Baan, and Computer Associates.

Manufacturing Resources Planning System

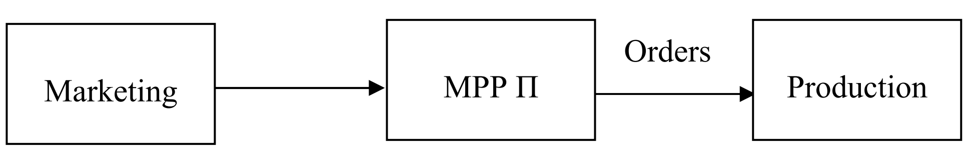


Figure 1 MRP Π Model

A primary benefit of using integrated information systems such as MRP Π is the improvement in the accuracy of information. Accurate and timely information is necessary to achieve lower production cost and higher customer service in today's complex supply chain. In a survey of MRP Π implementation and benefits, nine issues were accurately measured before and after the MRP implementation [10]. Table 1 presents benefits of the MRP Π implementation.

Table 1: BENEFITS ACHIEVED FROM MRP Π IMPLEMENTATION

Areas of Achievement
Improved market forecasts
Improved productivity
Better customer service
Reduced inventory costs
Improved competitive position
Better meeting of delivery date
Better manufacturing planning and control
Better production scheduling
Improved bill of material
Better shop floor control
More accurate cost estimation
Improved cooperation among departments
Improved product quality
Improved ability to meet volume/product changes

The inclusion of an optimization tool to aid in the planning or scheduling task of the MRP Π system can provide enormous business benefits. These benefits include: improved customer service, reduced inventories, reduced production costs, and greater flexibility.

This research proposes the use of the equations of Stefan Voss to develop a model that schedule jobs as late as possible without violating capacity constraints. This model was proposed by Prof. Voss who holds degrees in Mathematics (diploma) and Economics from the University of Hamburg and a Ph.D. and the habilitation from the University of Technology Darmstadt. Previous positions include full professor and head of the department of Business Administration, Information Systems and Information Management at the University of Technology Braunschweig (Germany), 1995-2002. Actually he is the Chair and Director of the Institute of Information Systems within the Faculty of Economics and Business Administration at the University of Hamburg (since 2002). The main focus of Prof. Voss interests is located in the fields of Information Systems, Supply Chain Management and Logistics for that reason Professor Voss developed a MRP Π model with optimization capability. The MRP Π model with optimization capability needs additional data when compared with the MRP model. The additional data requested is included in Table 2.

Table 2: DATA FOR A MRP Π FORMULATION

Data for a MRP Π Formulation
Number of SKUs
Number of time buckets
Number of resources
Beginning inventory of SKU i
Lead time for a lot of SKU i
Amounts of SKU i needed to make one j
External demand for SKU i in period t
Fraction of resource k needed to make one unit of SKU i
A random number; e.g., 1+1/(smallest U)

MRP Π requires computers with good speed and storage capability to handle the volumes of data and calculations required. This aspect becomes a drawback for the MRP Π optimization model. It is more difficult to solve a manufacturing resources planning problem using an optimization model of MRP Π, because we need more computer capacity and computational time.

For most manufacturing systems with large number of machines and many jobs with various routings competing for the various resources, an algorithmic solution to the scheduling problem is not possible. In these instances heuristics or 'rules of thumb' are often used for scheduling. These rules of thumb evolved over time through trial and error and are based on past experience of what have worked. For large scheduling problems 'the best' solution cannot be found within real-world time constraints. Therefore, the heuristic approach is to develop a schedule based on experience, which will work and will also be better than a random or unplanned schedule.

The problem of the requirement of computer capacity and computational time can be solved using a heuristic. Heuristic is the art and science of discovery and invention. The word comes from the same Greek root as "eureka", which means, "I find". A heuristic program design provides a framework for solving the problem in contrast with a fixed set of rules (algorithmic) that cannot vary.

Most of the research in heuristic methods demonstrates the potential of the heuristic to provide a good solution without the requirements of the optimization model. The heuristic does not guarantee optimality, but in many situations it does return the optimal solution.

The objective of this investigation is to develop a heuristic method to solve material resource planning problems. This solution will be compared to the results obtained from an optimization algorithm. The optimization algorithm will be developed based on the equations proposed by Voss [25]. The benefit of use these equations is that we can schedule jobs as late as possible without violating capacity constraints.

1.1 Justification

The idea for this research came out after taking a course that combines the linear programming solver with visual basic programming for supply chain and production planning. The use of an effective material resources planning system is very important for any business to compete on the global market [17]. But, what can be defined as an effective system? The system can be labeled as effective if it can provide good solutions on a shortened period of time. Voss [25] suggests a model we can provide an optimal solution. Since it uses integer variables, it will take a considerable computational time. The use of only continuous variables instead of integer will solve this problem.

Making some modification to Voss equations we can have a continuous-variable problem that helps in reducing the computational time. However, we did not know for sure how much time would be saved using the heuristic method. After an extended literature review, heuristic methods tackling the production scheduling with resource constraints and changeover considerations were not available, especially with the promise of short computational times.

1.2 Contribution

Every company wants to have an effective production planning and inventory control system. With the implementation of a realistic material resources planning model, companies can achieve many benefits besides improvements in data accuracy and inventory turnover. Some of these benefits are the following: better manufacturing planning and control, improved bill of material, reduced inventory cost, better production scheduling, more accurate cost estimation, and reduce the setup cost.

A material resources planning optimization model that takes into account resource constraints and changeover requirements will help companies with multiproduct and multitask production lines in defining a realistic production schedule that can be run frequently given the computing time requirements..

Using classical MRP Π software, a problem with few dozen time buckets and a few thousand SKUs is just too big to be solved easily. This research will develop a heuristic method for the material resources planning problem and will compare its results with the results of an optimization algorithm. It is important to observe the computational time requirement of each method to justify the need for the heuristic approach.

1.3 Objectives

The objectives of this research are:

1. To develop a computational time-efficient heuristic method for production scheduling with resource constraints and changeover considerations.

2. To program and execute the optimization model that will be used as benchmark to evaluate the appropriateness of the heuristic results.

3. To design and execute statistical experiments to determine how much time is required to obtain a solution in both, the full optimization model and the heuristic method, for varying problems sizes.

Chapter 2: Literature Review

There are several areas that have being studied for this research. One aspect is the evolution of production planning tools. It is very important to understand this evolution to identify problems on the different methods. Identifying these differences will help us develop a new method that takes into account all the required aspects in order to obtain a realistic solution.

In the past hundred years, production management has evolved from a set of heuristic ideas to a portfolio of somewhat developed concepts and principles. From Material Requirement Planning (MRP), developed in the early 1970, to Material Resources Planning defined in 1980s, these systems later evolved into the Enterprise Resource Planning (ERP). By examining the history, it could be inferred that the concept of ERP has evolved from simple inventory management systems of the 1960s to MRP systems in the 1970s and MRPΠ systems in the 1980s. The need for software specifically designed for manufacturing operations led to the development of MRP, MRPΠ and subsequently, ERP emerged in the 1990s [9].

2.1. Planning tools evolution

Santos et al [17] studied the evolution of management theory. They found that in general newer production theories do borrow concepts and principles already developed from previous work and therefore, usually generate only a fraction of truly new knowledge. Often the main body of these new theories is only a re-interpretation and re-configuration of previous theories in order to allow their application in a particular context and for particular a problem.

The second area that was investigated is the material requirement planning system. MRP originated in the 70's from a simple inventory control system. Initially, MRP was limited to the factory materials and planning, however on 1996 MRP appears as a simulation tool, which allows managers to examine the consequences of their production planning decisions.

2.2. Material Requirement Planning

Seyed-Mahmoud [17] investigated the contribution of material requirement planning system to company profitability. Although MRP systems have been in existence for almost 25 years, they have had their share of problems and are both challenged and enhanced by new supply chain management techniques and enterprise resource planning systems. As the World Wide Web (www) evolves into a global market of information, the information provided by MRP systems, will be necessary in order for businesses to compete in the global marketplace. Nowadays, in business climate, it is important to keep a tight control over inventories, including raw materials, work in process and finished goods. The ability to control inventories at various levels is a function of the control exerted over the inputs that comprise those same inventories. Material requirements planning is a system that attempts to decide on materials needs. It is a technique that is based around the concept of dependent demand. The concept of dependent demand states that the demand for one item is dependent on the demand for another item. These items are complimentary and one may require the other in order to function. A prime example of dependent demand would be auto tires and automobiles. If the demand for automobiles falls, the need for auto tires will decrease, leading to decreased demand for auto tires. When the demand is dependent, it is then possible to forecast the demand for the product and the quantities of materials needed to produce the final product. All components and subassembly requirements can be determined once dependency is established.

MRP Π is recognized as being an effective information management system that has an excellent planning and scheduling capability which can offer dramatic increases in customer service, significant gains in productivity, much higher inventory turns, and greater reduction in material costs. Owing to these benefits, the MRP Π systems have become one of the most rapidly growing computerization areas in the manufacturing sector.

2.3. Material Resources Planning

Lau et.al [11] identify factors that have an effect on the perceived benefits achieved from MRP Π implementation. When top management supports the implementation and has a clear goal for implementation, companies can achieve a higher degree of perceived benefits than companies that did not receive top management's support. Companies that implement the system according to a formal plan can achieve a higher degree of benefit than companies that do not develop a plan (or that perform the implementation by trial and error). Companies using continuous flow and job shop types of processes can expect a higher degree of perceived benefits from MRP Π implementation than companies using assembly line and batch types of processes. More involvement from people in other functional areas during the implementation process can serve as a good indicator of a successful MRP Π implementation.

W.H. Lp [20] develop a methodology to integrate a manufacturing strategy. This methodology describes the steps to be used, who should be involved, what information is needed, and what the outputs are. The process should be simple enough so that it can be easily followed. The company has identified its process for developing a manufacturing strategy; essentially it follows four major phases:

1. Establish the present position;
2. Analyze the strategic requirements;
3. Develop strategic improvements; and
4. Formulate the implementation strategy.

Barker J.R. [3] study the integration of material resources planning with real time interactive scheduling into schedule-based manufacturing (SBM). SBM requires considerably less manual intervention than MRPΠ. Production staffs are no longer required to do repetitious manual calculations, provide estimates and constantly adjust parameters required by MRPΠ. The computer resources (hardware and computer staff) required for SBM are considerably less than those required for MRPΠ. Software maintenance is also simplified by the absence of the traditional MRP, MPS, capacity planning and other associated functions.

Trigeiro, et al. [19] the effect of setup time on lot sizing. A Lagrangian relaxation of the capacity constraints of CLSP (Capacitated Lot Sizing Problem) allows it to be decomposed into a set of un-capacitated single product lot sizing problems. The Lagrangian dual costs are updated by sub gradient optimization, and the single-item problems are solved by dynamic programming. A heuristic smoothing procedure constructs feasible solutions which do not require overtime. The algorithm solves problems with setup time or setup cost. Problems with extremely tightly binding capacity constraints are much more difficult to solve. Solutions without overtime could not always be found for them. The most significant results are that the severity of the capacity constraint is a good indicator of problem difficulty for problems with setup time; and the algorithm solves larger problems better than smaller problems, although they are more time consuming to solve.

Absi, et al. [1] study a mixed integer mathematical formulation to solve problems for multi item capacitated lot sizing with setup time and shortage cost. Demand cannot be backlogged, but can be totally or partially lost. Safety stock is an objective to reach rather than an industrial constraint to respect. They also describe fast combinatorial separation algorithm for valid inequalities based on a generalization, using them in a branch and cut framework to solve the problem. The valid inequalities were generalized to take into account other practical constraints that occur frequently in industrial situations, notably minimal production level and minimal run constraints.

Porter, et. al. [15] study the production planning and control system development in Germany. Some manufacturing organizations, notably in the process sector (where bills of materials are generally not complex), will move towards finite capacity scheduling systems at shop-floor level, integrated into a host system which is itself a finite capacity scheduler capable of longer term planning and containing all the functionality of MRPΠ. Whether this is a better way of integrating the order chain from the forecast and/or order to planning to shop-floor scheduling depends on the nature of the manufacturing environment. Complex product environments, especially where synchronization of activities is important, may be better served by constraint based software which itself must have the associated database either from an MRP system or within its own logic.

After material resources planning system a new system have evolved on the 1990s, the enterprise resources planning system. Enterprise resources planning (ERP) system have been a popular information technology in the changing business environment [7]. ERP advocates believe that ERP combines both business processes in the organization and organizational information technology into one integrated solution.

2.3.1. Enterprise Resource Planning

Chung and Snyder [8] discusses a theoretical framework, MRP, MRPΠ, ERP, a summary on task and technology compatibility and propositions, and provide conclusions and directions for further research. Table 3 shows task technology integration in MRP, MRP Π and ERP. They found that ERP software still requires many resources and efforts to integrate all of the major business functions in the initiating firm. Those recommend a series of case studies and empirical tests on ERP for corporations with various stages of implementation.

Table 3: TASK-TECHNOLOGY INTEGRATION IN MRP, MRPΠ AND ERP

Technological context	Degree of potential Integration		
	MRPΠ	MRPΠ	ERP
Bill of materials	Low	High	High
Master planning schedule	Low	Medium	High
Capacity resource planning	Low	Medium	High
Value chain activities	Low	Medium	High
Customer demand forecast	Low	Low	High
Product development methodology	Low	Low	High
Data management	Low	Medium	High
Process repository	Low	Medium	High
IT connectivity	Low	Medium	High

Choosing the right ERP system for a company is key for gaining the competitive edge [5]. Supply chain software generally falls into one of two categories – ERP applications from companies like SAP, Baan and Oracle; and planning engine applications that support and integrate flow-based processes such as shop-floor, logistics, and inventory management. The following literature review presented the concept of supply chain management.

2.3.2. Supply Chain Management

Boubekri [5] focuses primarily on technology as the key enabler to improve supply chain management. Corporations seeking to improve their operation must look beyond the traditional cost-cutting approaches and focus on improving their overall supply chain. The emphasis lies on integrating their demand, supply, manufacturing/scheduling, transportation, and network optimization functions. Key technology such as ERP system enables to integration of these functions. This technology with the proactive involvement of top management should prove to be a significant differentiator in the quest for a more competitive position in the marketplace.

Understanding the production planning tool evolution and the concepts inside this evolution, we can change our attention to planning optimization. Optimization is an activity that has existed even longer than supply chains have. What is relatively new is the explosion in supply chain optimization software. In order to truly understand the material resources planning model and its proper role, we need to step back and understand optimization and how it can be applied to supply chains.

2.4. Optimization models in production planning

Peterson and Silver [22] provide optimal decision rules for inventory management and production planning to be able to display some of the more interesting, complex problems of individual item control where, in some cases, additional applied research is needed to develop truly operational decision systems.

Winston [24] describes the developments of material requirement planning. Material requirement planning recognizes the relationship between the demand for the final product and the components use to make it. Winston presents an optimization model for MRP and the restriction that can be present in these type of problems.

Taha [24] present two of models for material requirement planning problem. The first model assumes no setup cost and the second model assumes the setup cost. The second model presents a general dynamic programming algorithm. Taha also presents a heuristic model in which the unit production costs are constant and identical for all the periods.

General Dynamic Programming Algorithm:

z_i= Amount ordered

D_i= Demand for period i

X_i = Inventory at the start of period I

K_i = Setup cost in period I

h_i = Unit inventory holding cost from period i to i+1

The associated production cost function for period i is:

$$C_i(z_i) = \begin{cases} 0, & z_i = 0 \\ K_i + c_i(z_i), & z_i > 0 \end{cases} \tag{1}$$

For simplicity they assume that the holding cost for period is based on end of period inventory, which was define as:

$$X_{i+1} = X_i + z_i - D_i \tag{2}$$

$$\min_{0 \le z_i \le D_i + x_2} \left\{ C_i(z_i) + h_i X_{i+1} + f_{i-1}(X_{i+1} + D_i - z_i) \right\}, \quad i = 2,3,\ldots,n \tag{3}$$

Voss [25] presents the middle activity in the least tactical production planning. Beginning with models that are simple, these are then extended to include more details. The material resources planning model is modified to add: (i) Base cost objective function over time, (ii) Extra capacity and (iii) Allowing tardiness. The MRP Π problem is given as follow:

P	Number of products
T	Planning horizon
I (p, t)	Shock of product p at the end of period t
B (p, t)	Backorder for product p at the end of period t
R (p,t-1)	Units of product p requires at period t-1, available at period p
H (p)	Handling cost per unit/period for product p
CB (p)	Backorder order cost per unit/period for product p
CO (k)	Overtime cost for resource k
t (m, p)	Time require for product p on process m

x (p, t)	Units of products p started at period t

s (m,p)	Set up time require for product p on process m

δ (p,t)Є{0,1}	1 if product p is make on period t

γ (m,p,t) Є{0,1} 1 if product p is the last at period t-1 and the first at period t on machine m

cp (m, t)	Capacity available on process m at period t

UC (m,t)	Not use capacity for resource m at period t

minimize: $$\sum_{p=1}^{P} \sum_{t=1}^{T} \left(H_p I_{p,t} + CB_p B_{p,t}\right) + \sum_{m=1}^{m=M} \sum_{t=1}^{t=T} CO_m O_{m,t} \quad (4)$$

subject to:

$$\sum_{t=1}^{t-LT(i)} X_{i,\tau} + I(i,0) - \sum_{\tau=1}^{t} (D(i,t) + \sum_{j=1}^{P} R(i,j) X_{j,t}) \geq 0 \qquad \text{i= 1, . . . , P, t= 1,. . . , T} \quad (5)$$

$$\sum_{i-1}^{P} [U(i,k) X_{i,t} + S(i,k)(\delta_{i,t} - \gamma_{i,k,t})] \leq 1 \qquad \text{t=1, . . . ,T, k=1, . . .,K} \quad (6)$$

$$\delta_{i,t-1} + \delta_{i,t} \geq 2\gamma_{i,k,t}$$

$$\gamma_{i,k,t} / M \leq U(i,k) \qquad \text{t=1, . . . ,T, k=1, . . .,K} \quad (7)$$

$$\sum_{i=1}^{P} \gamma_{i,k,t} \leq 1$$

$$X_{i,t} \geq 0 \qquad \text{i= 1, . . .,P, t=1, . . . ,T} \quad (8)$$

Chen and Wang [7] developed a linear programming model to formulate the production and transportation planning problem based on the company's system structure and production practice. The model was illustrated by a smaller sized example and tested by large sized realistic problems. Critical analysis was conducted to obtain in-depth knowledge of the system. More profit could be achieved under the current internal and external production conditions. Cross-functional operations can be optimized and overall optimality can be obtained.

Model presentation

In presenting the mathematical programming model, the following notations are used.

i = index of factories, i Î{0, …, I}. i = 0 refers to the central plant;

j = index of raw material supplying territories, j Î{1, …, J};

k = index of territories of semi-finished product purchasing, k Î{1, ..., K};

l = index of customer regions, l Î{1, …, L};

m = index of product groups, m Î{1, …, M}; and

nm = index of product items in group m, n1 Î {1, …, Nj}, …, nm Î {1, …, Nm}.

The complete linear programming model can be expressed by:

$$MaxZ = \sum_{l=1}^{L}\sum_{m=1}^{M}\sum_{n=1}^{N_m} RP_{l.nm} \cdot \sum_{i=0}^{I}(X_{il,\,nm} + Y_{il,\,nm}) - \sum_{j-1}^{J} RC_j \cdot w_j - \sum_{i=0}^{I}\sum_{l=1}^{L}\sum_{m=1}^{M}\sum_{n=1}^{N_m}\{(FC_{nm}^{c} + FF_{nm}) \cdot X_{il,nm} + FF_{nm} \cdot Y_{il,nm}\} -$$

$$\sum_{i=0}^{I}\sum_{l=1}^{L}\sum_{m=1}^{M}\sum_{n=1}^{N_m}\{(VC_{nm}^{c} + VF_{nm}) \cdot X_{il,nm} + VF_{nm} \cdot Y_{il,nm}\} - \sum TS_i^c \cdot \sum_{l=1}^{L} \sum\sum \frac{X_{il,nm}}{YS_{nm}} - \sum_{i=0}^{I}\sum_{k=1}^{K} TS_{i,k}^{f} \cdot \sum_{m=1}^{M}\sum_{n=1}^{Nm} u_{ik,nm} -$$

$$\sum_{i=0}^{I}\sum_{l=1}^{L} TF_{ij} \cdot \sum_{m=1}^{M}\sum_{n=1}^{N_m}(x_{il,nm} + y_{il,nm}) \qquad (9)$$

Subject to:

$$w_j \le L_j, \forall j$$

$$\sum_{i=0}^{I}\sum_{l=1}^{L}\sum_{m=1}^{M}\sum_{n=1}^{N_m} \frac{x_{il,nm}}{YR_{nm} YS_{nm}} \le \sum_{j=1}^{J} w_j \qquad (10)$$

$$\sum_{i-0}^{I}\sum_{l=1}^{L}\sum_{m=1}^{M}\sum_{n=1}^{N_m} \frac{x_{il,nm}}{YR_{nm} \cdot PS_{nm}} \le \tau^{c} \qquad (11)$$

$$\sum_{l=1}^{L} \frac{y_{il,nm}}{YS_{nm}} \le \sum u_{ik,nm}, \forall i, \forall n, \forall m \qquad (12)$$

$$\sum_{l=1}^{L}\sum_{n=1}^{N_m} \frac{(x_{il,nm} + y_{il,nm})}{YS_{nm}} \le \tau_{im}^{f}, \forall i, \forall m \qquad (13)$$

$$\sum (x_{il,nm} + y_{il,nm}) \le DF_{l,nm}, \forall l, \forall n, \forall m \qquad (14)$$

$$x_{il,nm}, y_{il,nm}, u_{ik,nm} \ge 0, \forall i, l, k, n, m \qquad (15)$$

Knowing the production planning tools and the optimization models, it is beneficial to understand the combination of both aspects. How the production planning tool combined with optimization can help the companies reach their objectives. Advanced planning is a systematic scheduling of workers, materials, and machines by using lead times, time standards, delivery dates, workloads, and similar data for the purpose of producing products efficiently and economically and meeting desired deliver dates. This is based on orders from customers, production capacities, often a demand forecast, and the desired inventory levels in the supply chain.

2.4.1. Advanced Planning and Optimizing

Gaglioppa et al. [9] consider the planning and scheduling of production in a multi-task/multi-stage batch manufacturing process typical of industries such as chemical manufacturing, food processing, and oil refining. They formulate the problem as a mixed integer linear program. They show that the formulation leads to an NP-hard problem with a large integrality gap. They introduce the notion of echelon inventory and use it to construct a family of valid inequalities. In addition they show that the formulation with the additional constraints leads to a significantly tighter linear programming relaxation and to much reduced solution times for the mixed integer linear program.

Koclar and Soral [10] developed a production planning model that considers the multi-item capacitated lot sizing and sequencing problem for a process industry-type environments featuring time consuming and sequence dependent setups, which necessitate the integration of the lot sizing and sequencing steps in the production plan(general lotsizing & scheduling problem-GLSP). Formulating a nonlinear mathematical model to decompose the problem into two parts for lot sizing and sequencing, respectively, they proposed an iterative procedure for the solution. Table 4 above presents the mathematical model and the parameters and decision variables.

Table 4: PARAMETER AND DECISION VARIABLES FOR THE GLSP MODEL

Parameters		Decision Variables	
ST_{ij}:	Setup time the transition from item i to item j.	X_{jn}:	Production quantity of item j in position n
SC_{ij}:	Setup cost for the transition from item i to item j.	W_{jn}:	Binary variable indicating whether there is
h_j:	unit inventory holding cost for item j.	δ_{ijn}:	$\begin{cases} 1, & \text{if setup is incurred for the transition from the production of item i in position (n - 1) to that of item j in the next position (n)} \\ 0, & \text{otherwise} \end{cases}$
CO_t:	cost of overtime in period t.		
CP_j:	unit cost of production for item j.		
d_{jt}:	demand of item j in period t.		
C_t:	capacity in period t.		
P_j:	unit processing time of item j.		
N_t:	number of position in period t.		
$F_t(L_t)$:	first (last) position in period t.	I_{jt}:	Inventory of item j at the end of period t.
M_j:	minimum batch size of item j.	O_t:	Amount of overtime used in period t.

Minimize:
$$\sum_i \sum_j \sum_n \delta_{ijn} SC_{ij} + \sum_j \sum_t h_j I_{jt} + \sum_j \sum_n CP_j X_{jn} + \sum_t CO_t O_t \qquad (16)$$

Subject to:

$$I_{jt} = I_{j(t-1)} + \sum_{n=F_t}^{L_t} X_{jn} - d_{jt} \qquad \forall t, j \qquad (17)$$

$$P_j X_{jn} \le C_t W_{jn} \qquad \forall t, j, n = F_t..L_t \qquad (18)$$

$$\sum_j \sum_{n=F_t}^{L_t} P_j X_{jn} + \sum_i \sum_j \sum_{n=F_t}^{L_t} ST_{ij} \delta_{ijn} \le C_t + O_t \qquad \forall t \qquad (19)$$

$$\sum_j W_{jn} = 1 \qquad \forall n \qquad (20)$$

$$\delta_{ijn} \ge W_{i(n-1)} + W_{jn} - 1 \qquad \forall i, j, n \qquad (21)$$

$$X_{jn} \ge M_j (W_{jn} - W_{j(n-1)}) \qquad \forall j, n \qquad (22)$$

$$W_{jn} \quad binary \ \forall j, n, other \ \ positive \qquad (23)$$

Muckstadt, et al. [14] examine a discrete-time, periodic-review production environment that assembles several hundred items and that possesses limited, perhaps random production capacity. The demand for a large subset of these items is highly erratic and extremely difficult, if not impossible, to predict accurately. Consequently, a coordinated production-inventory strategy, such as the No B/C Strategy presented in Carr et al. is necessary [5].

In such a strategy, inventory is carried only in high demand rate, predictable items and production priority is given to non stocked items. Production is controlled for stocked items through a modified base stock policy. A key feature of this approach is that it does not rely on item-level forecasts for each item. The objective is to develop and test a computationally efficient and accurate procedure for establishing base-stock levels that minimize the expected holding and backorder costs per period over an infinite horizon.

Solving production planning problems for companies with multi-products and multitask require to much time and effort. Managing large amounts of information and efficiently using this information in improved decision making has become increasingly challenging as businesses collect terabytes of data [13]. Intelligent solutions, based on neural networks (NN) and genetic algorithms (GA), to solve complicated practical problems in various sectors are becoming more and more widespread nowadays.

2.4.2. New approaches to optimization

Metaxiotis and Psarras[12] provide an overview for the operations research of the neural networks and genetic algorithms methodology in business. Appearing from seemingly out of nowhere, neural network (NN) and genetic algorithms (GA) have quickly evolved from an academic notion into proven and highly marketable products. They provide powerful and flexible means for obtaining solutions to a variety of problems that often cannot be dealt with by other tools. The benefits reported from the use of NNs and GAs in business include more accurate decisions, time gains, flexibility, improved quality, effective training and minimization of human inconsistencies. Researches have shown that NNs and especially GAs have better performance than heuristics in large problems (from six to ten percent) and near to optimal in small problems.

The question stands concerning the gap of neural network and genetic algorithms with respect to the optimum in large problems. This aspect drives us to the last area of study: a heuristic model to solve material resources planning problems.

2.5. Heuristic models in production planning

Qui, Petterson and Cao [16] explores the constraints management portion of the heuristic model of Goldratt [21] to develop a production schedule for a set of machines in a complex manufacturing environment. They compare the solution obtained using the heuristic with a solution using mixed integer lineal programming. This comparison demonstrates the potential of the heuristic to provide a good solution without the requirements of the optimization model. The model is show in Table 5.   The heuristic does not guarantee optimality, but in many situations it does return the optimal solution.

Kenneth R. Baker [4] examines heuristic solution procedures for scheduling jobs on a single machine to minimize the maximum lateness in the presence of setup times between different job families. He reviews the state of knowledge about the solution of this problem, which is known to be difficult to solve in general, and examine natural solution approaches derived from some of the underlying theory. The method that he examine was the following:

Table 5: HEURISTIC'S FIRST THREE STEPS

Heuristic's First Three Steps

Step	Action
1. Identify constraint	a. We know that the spinning machines are the constraint.
2. Exploit the constraint	a. Find most profitable products for each machine. b. Make an initial schedule based on profitability c. See if adjustments to the initial schedule offer improvement. d. Finalize the schedule.
3. Subordinate to the constraint schedule	a. Use the constraint schedule to plan for feeding department b. Add safety by establishing a buffer to protect the constraint schedule. Connect the release of material to the production rate of the constraint. c. Manage using the buffer.

a. The Earliest Due Date Sequencing- when there is no setup time between jobs.

$$d_{[1]} \le d_{[2]} \le d_{[3]} \le \ldots \le d_{[n]}$$
$$where\ d - due\ date \quad [i] - position\ \ of\ \ i$$

Mingyun Chen [13] develops an inventory control model with production planning in order to minimize inventory and storage cost in cellular manufacturing systems. While cellular manufacturing analysis mainly addresses how machines should be grouped and parts be produced, a mathematical programming model was developed using an integrated approach for production and inventory planning. The mathematical programming model minimize inter-cell material handling cost, finish goods, inventory cost, and system setup cost. The non linear mixed integer programming model cannot be directly solved for real size practical problem. A decomposition based heuristic algorithm was then develop to efficiently solve the integrated planning and control problem.

Model parameters and coefficients are:

t = time index, $t= 1,\ldots,T$

i = part-type index, $i= 1,\ldots,I(t)$

j = index of operations part-type $i,j = 1,\ldots,J_i$

k = machine index, $k = 1,\ldots,K$

l = cell index, $l = 1,\ldots,L$

$H_i(t)$ = unit inventory holding cost of part-type i for time period t

$D_i(t)$ =known demand of part-type i for time period t

$M_k(t)$ = unit machine operating cost for machine type k in time period t

S_i = setup cost to produce part-type i

R_i = unit cost to move part-type i in batches between cells.

LB_l = minimum number of machines in cell l.

UB_l =maximum number of machines in cell t

Continuous decision variables

$X_i(t)$ = amount of part-type i to be processed in time period t

$v_i(t)$ = amount of part-type i in inventory at the end of time period t

0-1 Decision variables

$$z_{il}(t) = \begin{cases} 1, \text{if part - type } \iota \text{ is processed in cell } l \text{ during time t;} \\ \\ 0, \text{otherwise} \end{cases}$$

$$\delta_{i[jk]l}(t) = \begin{cases} 1, \text{if operation j of part } \iota \text{ to be processed by machine k} \\ \text{is dode in cell l during time t;} \\ \\ 0, \text{otherwise} \end{cases}$$

$$\beta_i(t) = \begin{cases} 1, \text{if part - type } \iota \text{ is processed in during time t} \\ \\ 0, \text{otherwise} \end{cases}$$

$$n_{kl}(t) = \begin{cases} 1, \text{if one unit of type k machine is placed in cell l at time t;} \\ \\ 0, \text{otherwise} \end{cases}$$

The mathematical model can be expressed as follows:

$$\text{MIN}\sum_{t=1}^{T}\sum_{i=1}^{I(t)} R_i x_i(t)[\sum_{l=1}^{L} z_{il}(t)-1]+\sum_{t=1}^{T}\sum_{k=1}^{K} M_k(t)\sum_{l=1}^{L} n_{kl}(t)+\sum_{t=1}^{T-1}\sum_{i=1}^{I(t)} H_i(t)v_i(t)+\sum_{t=1}^{T}\sum_{i=1}^{I(t)} S_i\beta_i(t) \quad (24)$$

subject to:

$$\sum_{l=1}^{L}\delta_{i[jk]l}(t)=\beta_i(t),\ j=1,...J_i, i=1,...,I, \qquad \forall t; \qquad (25)$$

$$\delta_{i[jk]l}(t)\le z_{il}(t),\ j=1,...,j_i, i=1,...,I, l=1,...L, \qquad \forall t; \qquad (26)$$

$$\delta_{i[jk]l}(t)\le n_{kl}(t), k=1,...,K, l=1,...L, \qquad \forall t; \qquad (27)$$

$$LB_l\le\sum_{k=1}^{K} n_{kl}(t)\le UB_l, l=1,...,L \qquad \forall t; \qquad (28)$$

$$v_i(t+1)=v_i(t)+x_i(t)-D_i(t), i=1,...,I, t=1,...,T-1 \qquad (29)$$

$$\sum x_i(t)=\sum D_i(t), i=1,...,I; \qquad (30)$$

$$\beta_i(t)=\begin{cases}1,\text{if } \mathrm{x_i}(t)>0, \\ \qquad\qquad i=1,...,I, \\ 0,\text{if } \mathrm{x_i}(t)=0,\end{cases} \qquad \forall t; \qquad (31)$$

$$x_i(t)\ge 0, v_i(t)\ge 0, \delta_{i[jk]l}(t), n_{kl}(t), z_{il}(t)=0,1, \forall i, j, l, t. \qquad (32)$$

Chang, Hasting and White [6] developed a fast production scheduling system, called the very fast scheduler (VFS). It schedules production at a rate exceeding 100 operations per second of elapsed time on an IBM-PC compatible computer with a 80486 processor. The speed of the VFS is such that practical production problems involving several thousand operations can be scheduled or rescheduled in less than one minute. The quality of the resulting schedules is comparable to, or better than, those produced by alternative heuristic scheduling techniques [10]. The VFS can be used interactively, allowing the user to redefine capacity available and derive a new schedule within one minute.

Basic Scheduling Technique:

The method used by the VFS involves combining the job-oriented heuristic (JOH) approach[1] with the use of a work group capacity scheduling technique.

JOH scheduling involves two stages:

(1) Sort the jobs into a loading sequence based on such factors as:

- technological precedence (e.g. table legs must be made before tables can be assembled);
- i due date; and
- i management priority.

(2) Load the jobs on to the available capacity in the sequence determined at

Step 1. The loading may be either forward from time now or backward

from due date. Daily time buckets are used in this phase.

From this literature review, there is no evidence of heuristic models that can solve large problems of material resources planning. To solve large problems of material resources planning the literature suggest an optimization algorithm, but the optimization algorithm need more computational time and capacity than heuristic methods. The intention of this research is to develop a heuristic method that can solve large problems of MRP Π with short computational time and near to optimum solutions.

Chapter 3: Methodology

This chapter presents the proposed methodology for this research. Figure 2 shows a flowchart of this methodology. The following section describes in detail what each step entails.

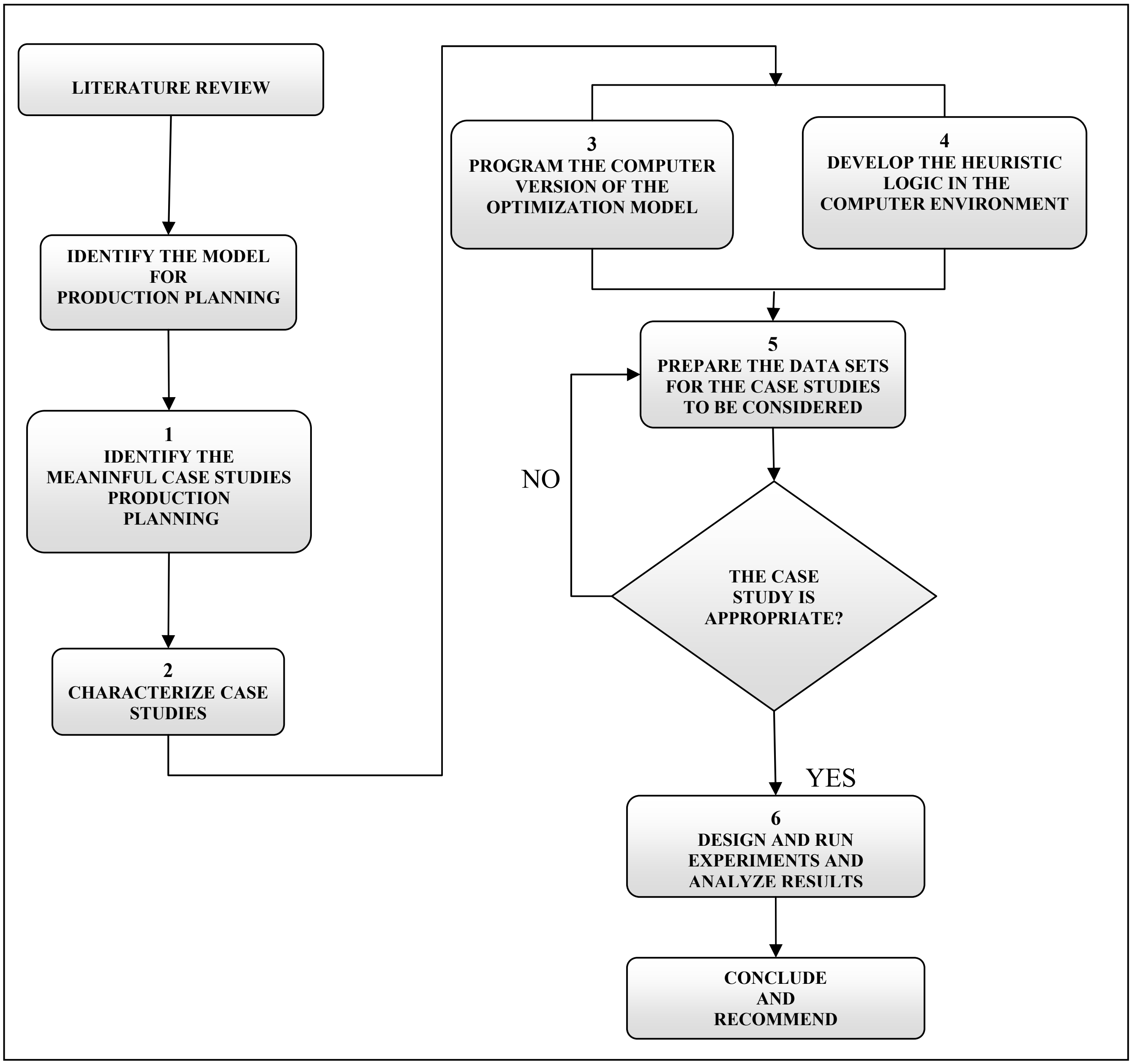


Figure 2: Methodology Flowchart

3.1 Detailed Description of Methodology Steps

After presenting the flowchart, this section will describe in detail the proposed methodology taking into account the literature review presented in Chapter 2 and the specific characteristics for the production planning model. The following steps describe the proposed methodology.

1. Identify meaningful production planning case studies to be used by the optimization and heuristic models. It is important to identify the areas of production planning that will be emphasized because the complexity of the problem depends on them.

2. Characterize case studies in terms of the size of bill of material, resource requirements and planning horizon. The problem needs to have between 30 and 40 products, including sub-assemblies and finished goods. The complexity of the problem will be determined by the number of products; as complexity increases, the effect of the heuristic approach will be observed.

3. Program the computer version of the optimization model to be used as benchmark for comparison with the heuristic results. The selected optimization model is the one presented by Voss [25].This computer version of the optimization model will be developed using visual basic. The programming equations are the following:

Minimize:
$$\sum_{p=1}^{P} \sum_{t=1}^{T} \left(HpIp,t + CBpBp,t \right) + \sum_{m=1}^{m=M} \sum_{t=1}^{t=T} COmOm,t \tag{33}$$

subject to:

$$I(p,t-1)-B(p,t-1)+X[p,t-LT(p)]-D(p,t)-\sum_{j=1}^{P}R(p,j)X(j,t)=I(p,t)-B(p,t) \tag{34}$$

p= 1, . . . , P, t= 1,. . . , T

$$\sum_{p=1}^{P}[t(m,p)*X(p,t)+s(m,p)*[\delta(p,t)-\gamma(m,p,t)]+UC(m,t)=cp(m,t) \quad (35)$$

t=1, . . . ,T,   m=1, . . .,M

$$\left.\begin{array}{l}\delta(p,t) \geq X(p,t)/M \\ \delta(p,t-1)+\delta(p,t) \geq 2\gamma(p,m,t) \\ \sum_{p=1}^{P}\gamma(p,m,t) \leq 1\end{array}\right\} \quad t=1,\ldots,T, \quad m=1,\ldots,M \quad (36)$$

X(p,t) non-negative   p= 1, . . . , P, t= 1,. . . , T   (37)

4. Develop the heuristic logic in the computer environment. This logic will be developed using visual basic and in LP solver. Figure 3 presents a flowchart of the heuristic logic. The formulation for the heuristic comprises of equations (33), (34) and (38) below, which is a simplified version of equation (35). The heuristic method will stop when we receive the same value for the objective function on two consecutive runs, or when we obtain a pattern on the run sequence.

$$\sum_{p=1}^{P} t(m,p)*X(p,t)+UC(m,t)=cp(M,t) \qquad t=1,\ldots,T, \quad m=1,\ldots,M \qquad (38)$$

5. Prepare various the data set required for each case study to be considered. It is important to evaluate the results obtained from the optimization to confirm that the data set has adequate complexity to require a relevant amount of time to reach the

optimum. Table 6 presents the data that is required as input for the Visual Basic program to build the objective function and constraints for the problem to be optimized.

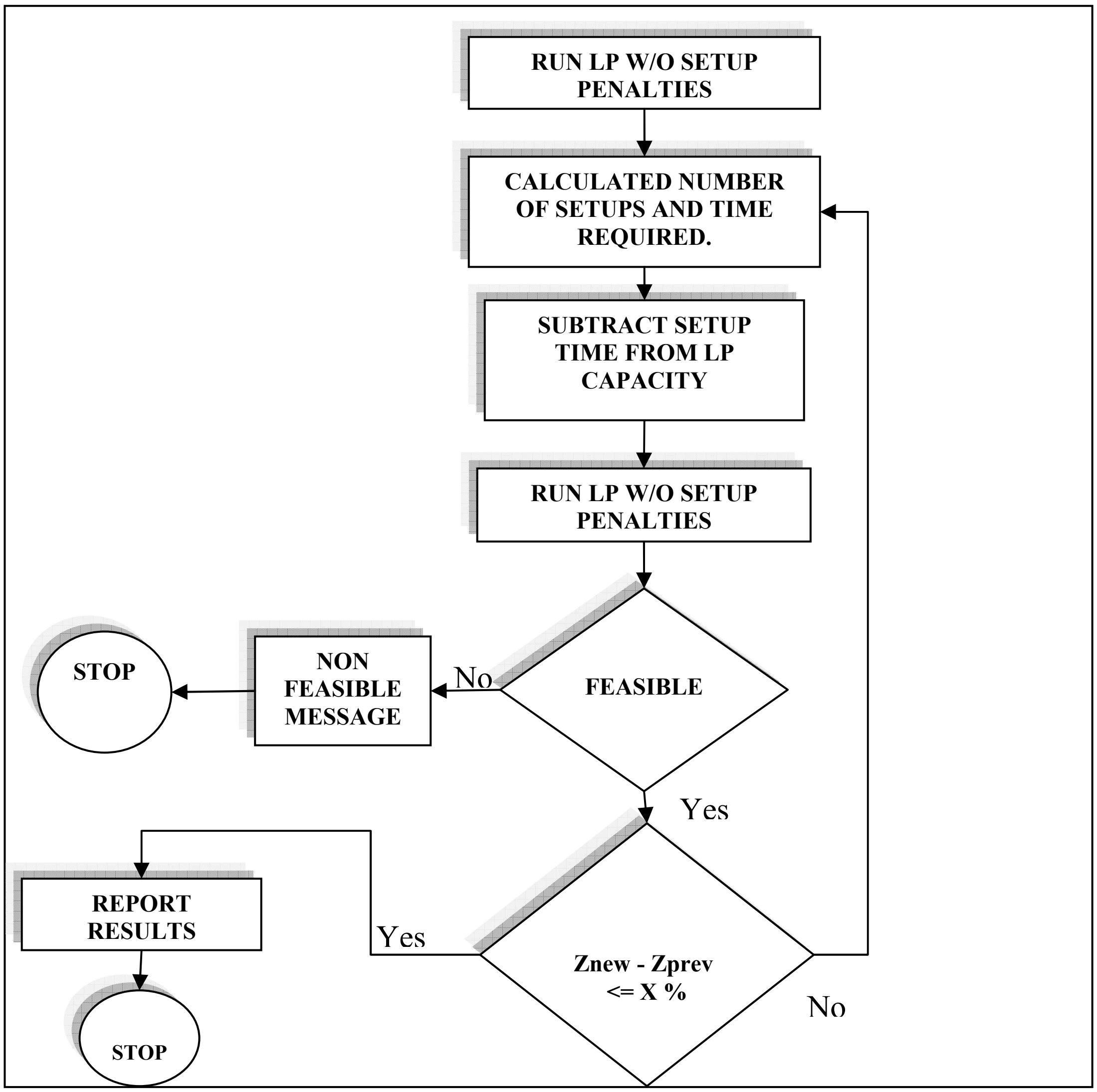


Figure 3: Heuristic logic flowchart

6. Perform runs aligned with an experiment of interest, comparison the optimization and the heuristic results in terms of objective function values and computational time requirements. Figure 4 presents the Excel worksheet in which the author initiates the

Table 6: INITIAL DATA REQUIRE BY MODEL

Product	Initial Inv	Final Inv	Lead Time	Lot Size	Inv Cost	BL Cost	Initial BL	yield	Demand=>	Periods 01	02	03	04	05	06	07	08	09	10	11	12
FL	200	0	1	1	5.0	400	0	1		100	100	100	100	100	100	100	100	100	100	100	100
Body	100	0	2	1	1.0	200	0	1		0	0	0	0	0	0	0	0	0	0	0	0
Switch	500	0	3	1	0.5	50	0	1		0	0	0	0	0	0	0	0	0	0	0	0
CS	200	0	2	1	1.0	200	0	1		0	0	0	0	0	0	0	0	0	0	0	0
Bulb	650	0	5	1	0.5	100	0	1		0	0	0	0	0	0	0	0	0	0	0	0
SoftLC	200	0	2	1	0.2	50	0	1		0	0	0	0	0	0	0	0	0	0	0	0
EmergLC	200	0	2	1	0.2	50	0	1		0	0	0	0	0	0	0	0	0	0	0	0
Rfl	500	0	3	1	0.3	30	0	1		0	0	0	0	0	0	0	0	0	0	0	0
Lens	1000	0	2	1	0.2	50	0	1		0	0	0	0	0	0	0	0	0	0	0	0
Support	800	0	4	1	0.2	30	0	1		0	0	0	0	0	0	0	0	0	0	0	0
TH	500	0	3	1	0.2	20	0	1		0	0	0	0	0	0	0	0	0	0	0	0
BC	500	0	3	1	0.3	20	0	1		0	0	0	0	0	0	0	0	0	0	0	0
PT	1000	0	5	1	0.2	30	0	1		0	0	0	0	0	0	0	0	0	0	0	0

needed computer runs, through the use colored buttons, which are linked to visual basic code.

7. Perform statistical analysis on the objective function gap between the optimization and heuristic results and the computational time requirements. With this analysis we will know how close the heuristic results are to the optimization model and how quickly the results are obtained

3.2 Case Study Developed as Part of this Research

A case study search was conducted with purpose of identifying relevant and well-known data sets that could be used to exercise the optimization and heuristic model. Contact was established with Trigeiro [19] and Absi [1]. However their case studies were too small and simple compared to the problem complexity desired. The author decided to develop a case study based on a flashlight. Figures 5 and 6 show the diagrams for the product. Tables 7 and 8 present the bill of material (BOM) for the two flashlights; the only difference between them is the BOM size and the process requirements. All the plastic parts will be produced by injection molding machines; for example, as shown in Figure 6 the emergency light cover and the body will be produced on the same machine based on the color of the part (orange and red parts). The soft light cover and the bulb support will be produced on another machine (white parts). The tail cap, the tail and head support and the o-ring head will be produced on another machine (black parts) for a total of three parallel machines and three operators for the assembly process on the production floor. All the electrical parts such as bulbs, switches, reflectors, clear lenses and electrical contacts will be purchased. Figure 4 shows a diagram of the production line.

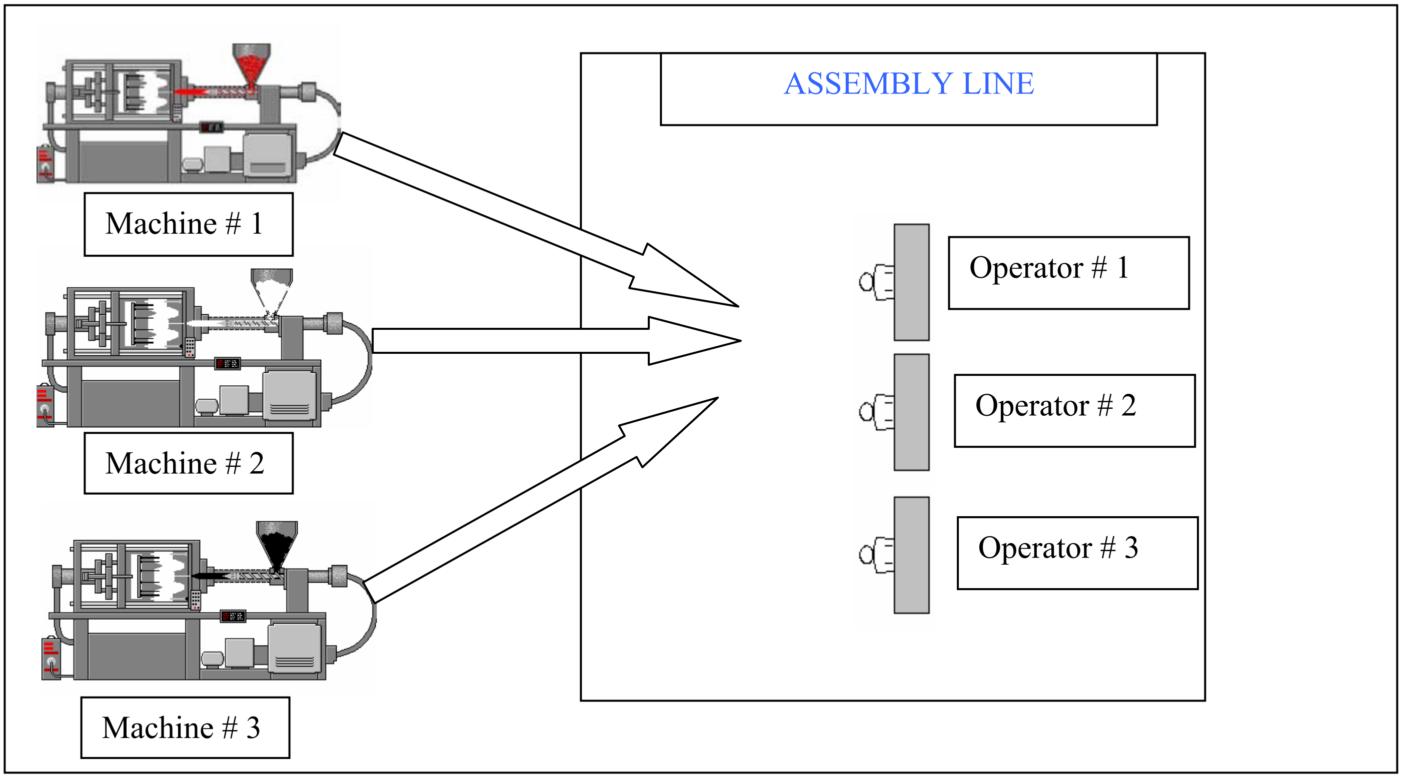
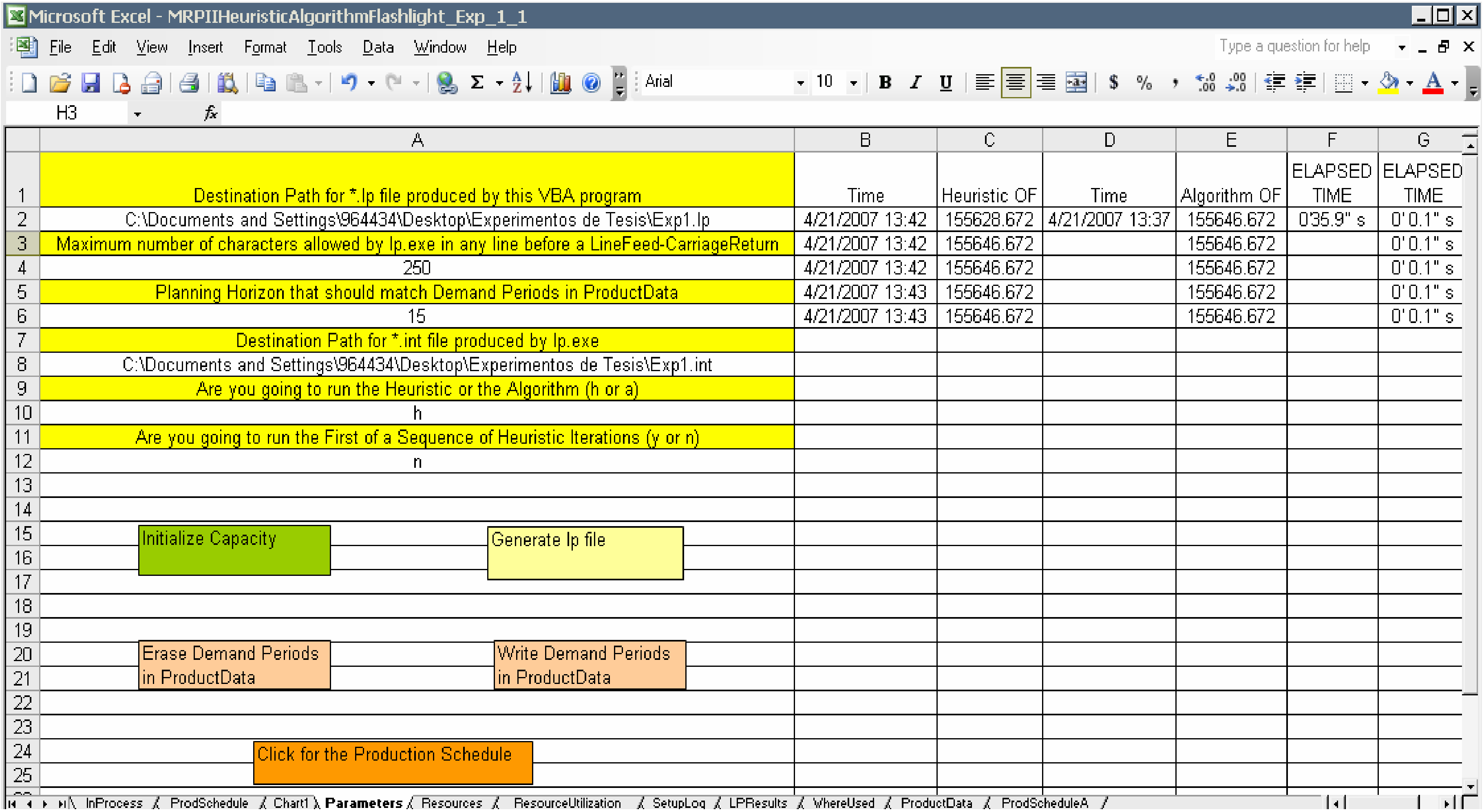


Figure 4: Production line diagram

For this cases study there are four factors that need to be taken into consideration: (1) the number of products, (2) the number of time periods, (3) the number of machines, and (4) the capacity utilization. Table 9 provides a detailed description on how these four factors were mixed in the experimental design of interest.


Microsoft Excel - MRPIIHeuristicAlgorithmFlashlight_Exp_1_1

	A	B	C	D	E	F	G
1	Destination Path for *.lp file produced by this VBA program	Time	Heuristic OF	Time	Algorithm OF	ELAPSED TIME	ELAPSED TIME
2	C:\Documents and Settings\964434\Desktop\Experimentos de Tesis\Exp1.lp	4/21/2007 13:42	155628.672	4/21/2007 13:37	155646.672	0'35.9" s	0'0.1" s
3	Maximum number of characters allowed by lp.exe in any line before a LineFeed-CarriageReturn	4/21/2007 13:42	155646.672		155646.672		0'0.1" s
4	250	4/21/2007 13:42	155646.672		155646.672		0'0.1" s
5	Planning Horizon that should match Demand Periods in ProductData	4/21/2007 13:43	155646.672		155646.672		0'0.1" s
6	15	4/21/2007 13:43	155646.672		155646.672		0'0.1" s
7	Destination Path for *.int file produced by lp.exe						
8	C:\Documents and Settings\964434\Desktop\Experimentos de Tesis\Exp1.int						
9	Are you going to run the Heuristic or the Algorithm (h or a)						
10	h						
11	Are you going to run the First of a Sequence of Heuristic Iterations (y or n)						
12	n						

Figure 5: Example of Workbook use to run the optimization model and the heuristic model

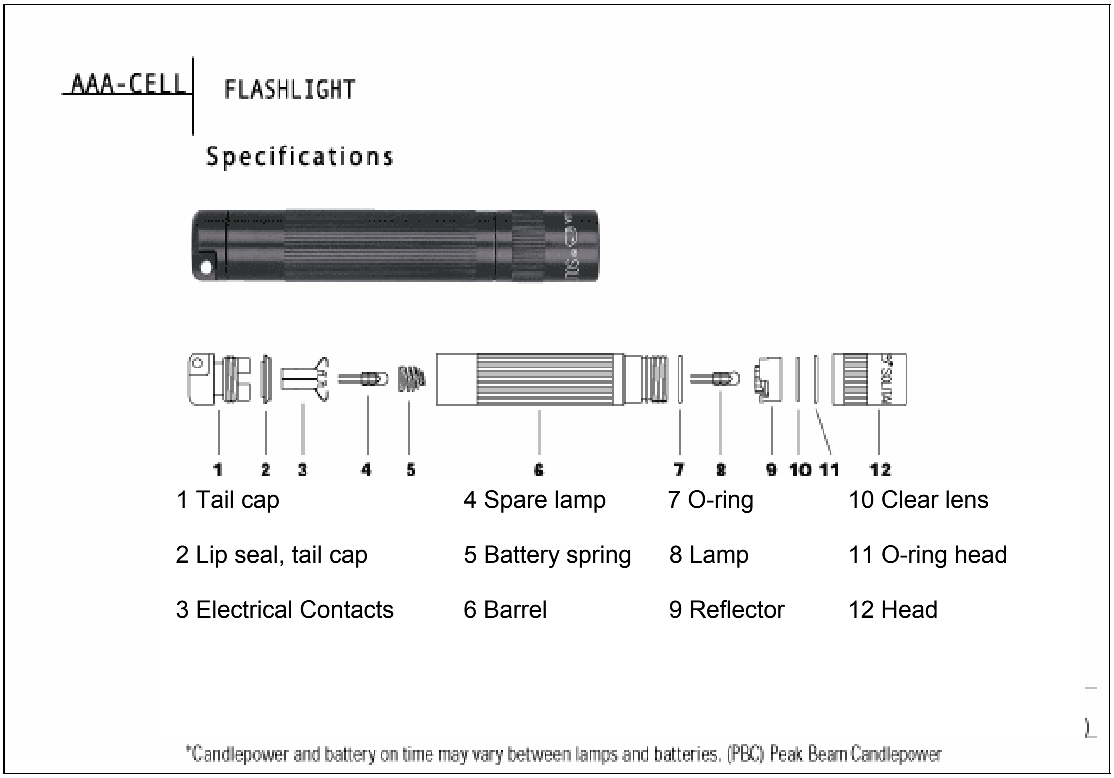


Figure 6: Regular Flashlight Diagram

Table 7: REGULAR FLASHLIGHT BOM

PART DESCRIPTION	QUANTITY ASSEMBLY	USED BY
Tail cap	1	Flashlight (finish product)
Lip Seal	1	Electrical contacts
Electrical Contact	1	Tail cap
Battery Spring	1	Tail cap
Barrel	1	Flashlight (finish product)
O-ring	2	Head
Lamp	1	Head
Reflector	1	Head
Clear lens	1	Head
Head	1	Flashlight (finish product)

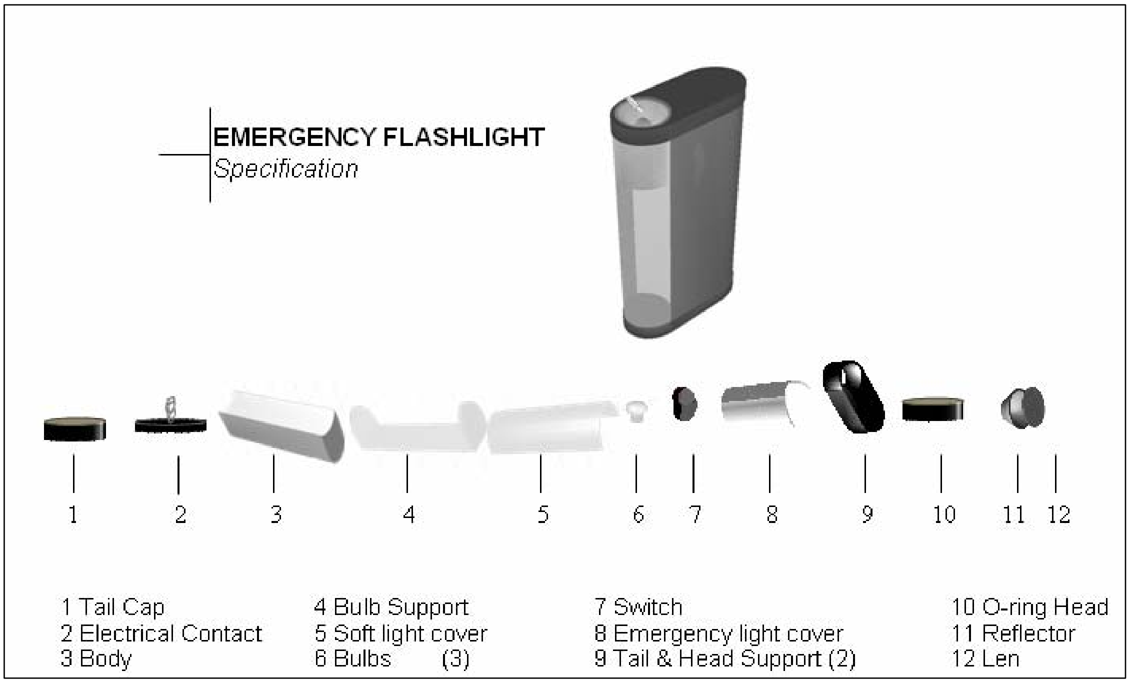


Figure 7: Emergency Flashlight Diagram

Table 8: EMERGENCY FLASHLIGHT BOM

PART DESCRIPTION	QUANTITY ASSEMBLY	USED BY
Tail cap	1	Emergency Flashlight
Electrical Contact	1	Tail cap
Body	1	Emergency Flashlight
Bulb Support	1	Body
Soft light cover	1	Emergency Flashlight
Bulbs	1	Bulb Support
Switch	1	Body
Emergency light cover	1	Emergency Flashlight
Tail & head support	2	Emergency Flashlight
Reflector	1	Emergency Flashlight
Clear len	1	Emergency Flashlight
O-ring Head	1	Emergency Flashlight

3.3 Cost Components for the Optimization and Heuristic Models

The objective of the optimization model is to minimize the holding cost, the backlog cost and the over time cost. As capacity is still available, backlog is avoided. When the regular time capacity is fully utilized, the optimization too will compare the cost of using overtime against the cost of creating backlog.

3.4 Detailed Description of the Constraints Used in the Models

The methodology considers understanding the constraints inside the optimization algorithm developed by Voss [25]. The first equation below is related to the material balance constraint:

$$I(p,t-1)-B(p,t-1)+X[p,t-LT(p)]-D(p,t)-\sum_{j=1}^{P}R(p,j)X(j,t)=I(p,t)-B(p,t) \quad \forall \quad p,t$$

The next equation limits the capacity consumption per machine type for $\forall\, m,t$:

$$\sum_{p=1}^{P}[t(m,p)*X(p,t)+s(m,p)*[\delta(p,t)-\gamma(m,p,t)]+UC(m,t)=cp(m,t)$$

And the next three constraints are related to setup time:

$\delta(p,t) \geq X(p,t)/M$ which triggers set-up time when a product is built;

$\delta(p,t-1)+\delta(p,t)\geq 2\gamma(p,m,t)$ to avoid the set-up time at the beginning of period t when the last product in period t-1 is the first in period t;

$\sum_{p=1}^{P}\gamma(p,m,t)\leq 1$ to limit the avoidance of set-up time to only one product per period.

Chapter 4: Results

As explained before, a statistical experiment was designed with the intent of measuring (1) the closeness of the objective function value and (2) the computational time requirements for the optimization model (algorithm) versus the heuristic approach developed as part of this research. The analyses of results are presented on this chapter. All computer runs were made in desktop Dell personal computers with a velocity of 3.5 giga-hertz (ghz).

4.1 Design of the Experiment to Compare the Algorithm and the Heuristic Approaches

A factorial experiment was designed, including four independent variables or factors at two levels each. Table 9 shows the factors used in the design, which is being evaluated with the Minitab software. The factors considered in the experiment are: (1) the number of final products which relates to the size of the bill of materials (BOM); (2) the number of periods or time buckets; (3) the number of machines in use; and (4) the closeness to full capacity utilization. The factors selected are relevant issues in the production floor which have a bearing on the size of the constraint set, which then impact the computational requirements for reaching a solution.

Each factor was considered at two levels. The first factor, the number of final products, had values of one and three products. The second factor, the number of time periods, had values of 15 and 20 periods. The third factor, the number of machines, had values of two and three machines. The last factor, closeness to full capacity, was considered at close to 80 percent versus between 90 and 95 percent. This latter factor is a result of the model run, whose value is achieved by adjusting the magnitude of the final product demand.

Table 9: FACTORS AND LEVELS DEFINED IN THE EXPERIMENTAL DESIGN DESIGN

Experimental condition	Number of Product **(A)**	Number of Periods **(B)**	Number of Machines **(C)**	Percent of Capacity Utilization **(D)**
1	1	15	2	80%
2	3	15	2	80%
3	1	20	2	80%
4	3	20	2	80%
5	1	15	3	80%
6	3	15	3	80%
7	1	20	3	80%
8	3	20	3	80%
9	1	15	2	90-95%
10	3	15	2	90-95%
11	1	20	2	90-95%
12	3	20	2	90-95%
13	1	15	3	90-95%
14	3	15	3	90-95%
15	1	20	3	90-95%
16	3	20	3	90-95%

The objective function values obtained for the cost minimization models are presented in Table 10, which includes 16 experimental conditions given the fact that a 2^4 experiment was performed. Columns six and seven contain the heuristic and algorithm objective function values, respectively.

Table 10: OBJECTIVE FUNCTION RESULT BY EXPERIMENTAL CONDITION

Experimental Condition	Number of Product **(A)**	Number of Periods **(B)**	Number of Machines **(C)**	Percent of Capacity Utilization **(D)**	Production Cost for the Heuristic	Production Cost for the Algorithm
1	1	15	2	80%	155,628.67	155,646.67
2	3	15	2	80%	2,016,326.87	2,016,322.88
3	1	20	2	80%	155,646.67	155,646.67
4	3	20	2	80%	2,016,322.88	2,016,326.87
5	1	15	3	80%	155,678.40	155,678.41
6	3	15	3	80%	2,016,570.62	2,016,568.00
7	1	20	3	80%	155,678.40	155,678.40
8	3	20	3	80%	2,016,568.00	2,016,567.00
9	1	15	2	90-95%	1,533,692.50	1,533,573.63
10	3	15	2	90-95%	3,189,789.00	3,189,671.00
11	1	20	2	90-95%	1,597,066.87	1,596,736.50
12	3	20	2	90-95%	3,189,664.75	3,189,564.75
13	1	15	3	90-95%	1,534,126.12	1,534,007.25
14	3	15	3	90-95%	3,190,379.00	3,190,226.00
15	1	20	3	90-95%	1597,813.12	1597500.00
16	3	20	3	90-95%	3,190,379.00	3,190,217.00

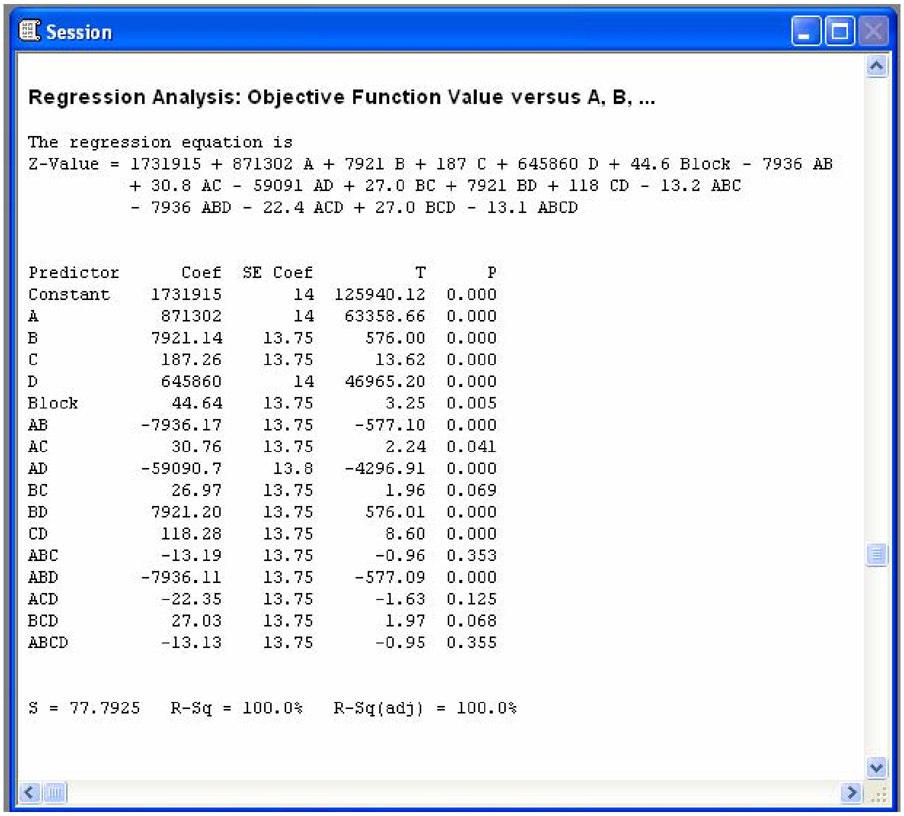
4.2 Results Obtained from the Experiment

For the analysis, the set of results belonging to the algorithm versus the heuristic approach were identified by using a block effect. Each block contains 16 or 2^4 results; thus, a total of 32 ($2x2^4$) values were used in the statistical analysis. The analysis was performed with the Minitab software.

4.2.1 Analysis of Results Considering Blocking

A linear regression model was used to evaluate the block effect, main effects, and two-, three-, and four- factor interactions. Using a confidence level of 95 percent ($\alpha = 0.05$), the regression model results presented in Figure 7 show that all effects are significant except two of the four three-factor interactions neither the four-factor interaction. Figure 8 presents the residuals analysis for the complete regression model. These results do not comply with normality or constant variance. However, this first model was used to identify which effects were relevant,

A second model was considered leaving out those non-significant high order interactions; i.e. ABC, ACD, BCD, and ABCD. The results of this model are presented in Figures 9 and 10. Now results are in compliance with normality and constant variance assumptions for residuals. The behavior of residuals can be related to the lack of spreading of the objective function values, presented in Figure 11. Four clusters contain all results; both the algorithm (points 1-16) and heuristic (points 17-32) show very similar results for the varying factor combinations. To validate that residuals are truly normal, a Kolmogorov-Sminov test was performed. Figure 12 shows a p-value of .15, for which there is no evidence to reject the normality claim.



Session

Regression Analysis: Objective Function Value versus A, B, ...

```
The regression equation is
Z-Value = 1731915 + 871302 A + 7921 B + 187 C + 645860 D + 44.6 Block - 7936 AB
          + 30.8 AC - 59091 AD + 27.0 BC + 7921 BD + 118 CD - 13.2 ABC
          - 7936 ABD - 22.4 ACD + 27.0 BCD - 13.1 ABCD
```

Predictor	Coef	SE Coef	T	P
Constant	1731915	14	125940.12	0.000
A	871302	14	63358.66	0.000
B	7921.14	13.75	576.00	0.000
C	187.26	13.75	13.62	0.000
D	645860	14	46965.20	0.000
Block	44.64	13.75	3.25	0.005
AB	-7936.17	13.75	-577.10	0.000
AC	30.76	13.75	2.24	0.041
AD	-59090.7	13.8	-4296.91	0.000
BC	26.97	13.75	1.96	0.069
BD	7921.20	13.75	576.01	0.000
CD	118.28	13.75	8.60	0.000
ABC	-13.19	13.75	-0.96	0.353
ABD	-7936.11	13.75	-577.09	0.000
ACD	-22.35	13.75	-1.63	0.125
BCD	27.03	13.75	1.97	0.068
ABCD	-13.13	13.75	-0.95	0.355

S = 77.7925   R-Sq = 100.0%   R-Sq(adj) = 100.0%

Figure 8: Regression Analysis for the Complete Model

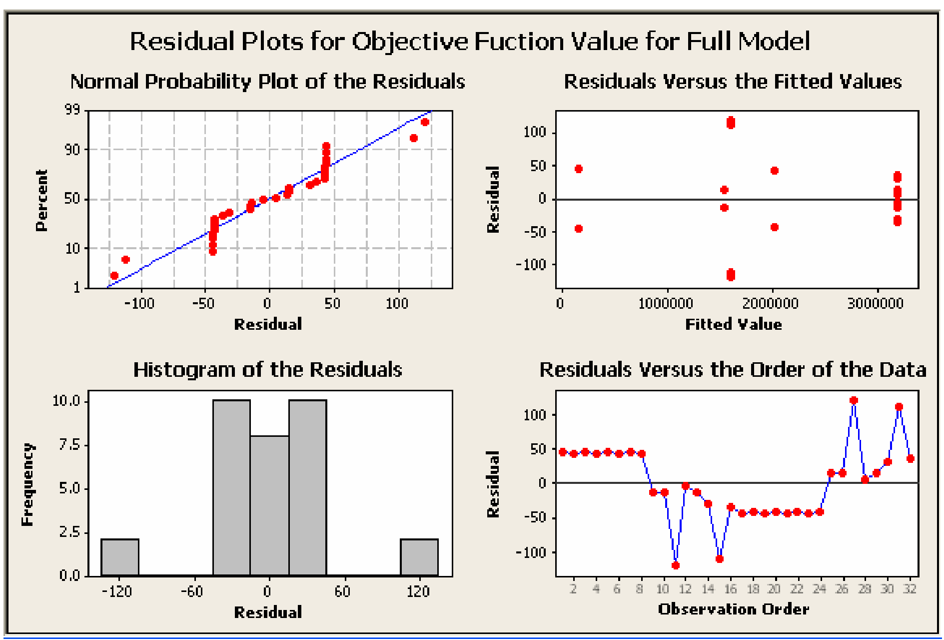


Figure 9: Residual Analysis for Full Model

Session

Regression Analysis: Objective Function Value Coinsidering Block

```
The regression equation is
ValorZ = 1731915 + 871302 A + 7921 B + 187 C + 645860 D
         + 44.6 Bloque - 7936 AB + 30.8 AC - 59091 AD + 27.0 BC + 7921 BD
         + 118 CD - 7936 ABD
```

Predictor	Coef	SE Coef	T	P
Constant	1731915	15	113638.08	0.000
A	871302	15	57169.68	0.000
B	7921.14	15.24	519.74	0.000
C	187.26	15.24	12.29	0.000
D	645860	15	42377.56	0.000
Bloque	44.64	15.24	2.93	0.009
AB	-7936.17	15.24	-520.72	0.000
AC	30.76	15.24	2.02	0.058
AD	-59090.7	15.2	-3877.18	0.000
BC	26.97	15.24	1.77	0.093
BD	7921.20	15.24	519.74	0.000
CD	118.28	15.24	7.76	0.000
ABD	-7936.11	15.24	-520.72	0.000

S = 86.2140   R-Sq = 100.0%   R-Sq(adj) = 100.0%

Analysis of Variance

Source	DF	SS	MS	F	P
Regression	12	3.77614E+13	3.14679E+12	4.23362E+08	0.000
Residual Error	19	141224	7433		
Total	31	3.77614E+13			

Figure 10: Regression Analysis for the Simplified Model

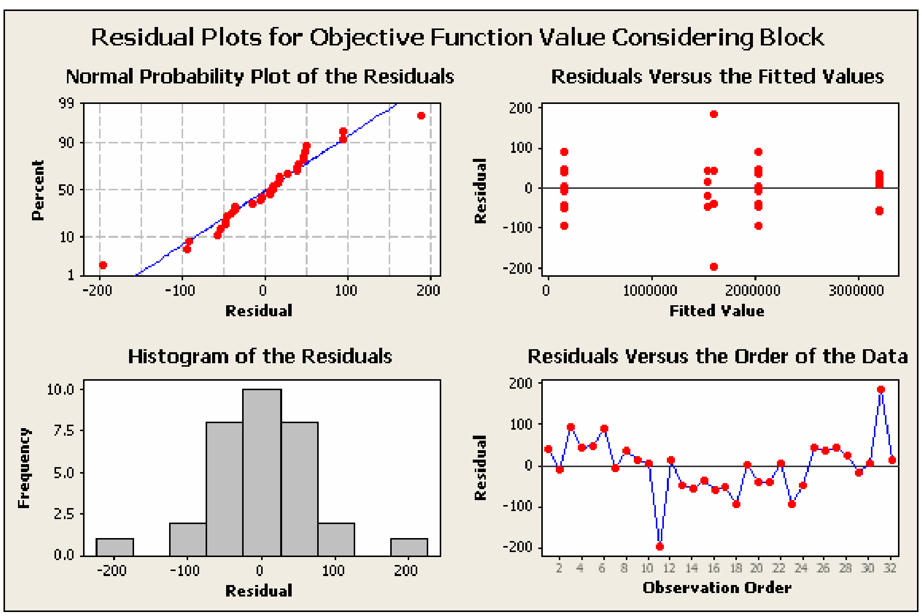


Figure 11: Residual Analysis for Simplified Model

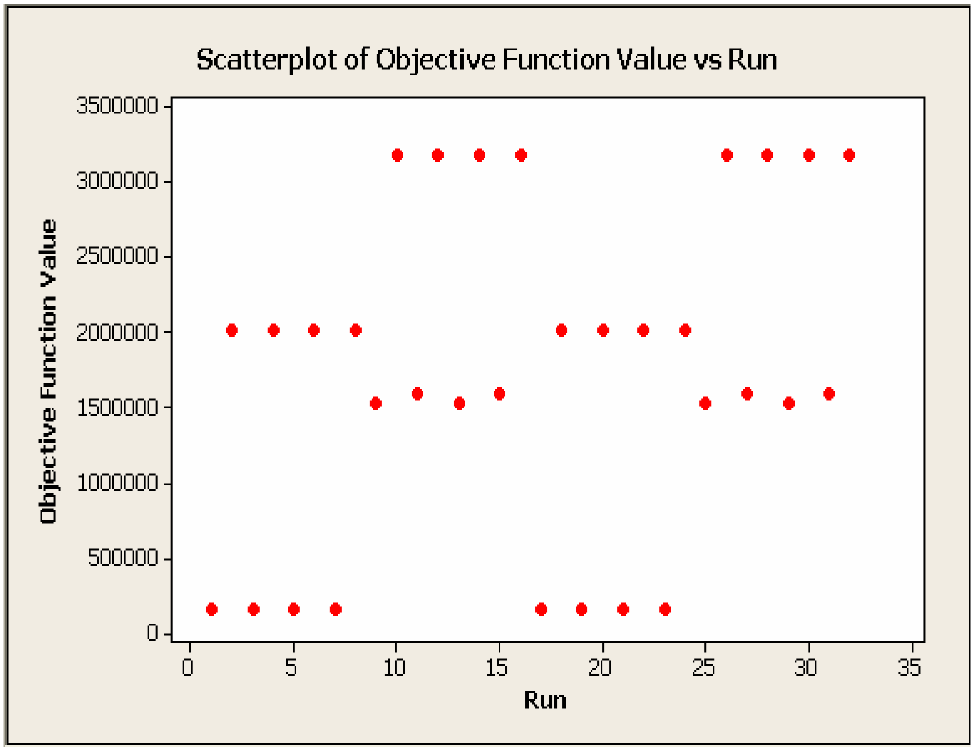


Figure 12: Objective Function Values

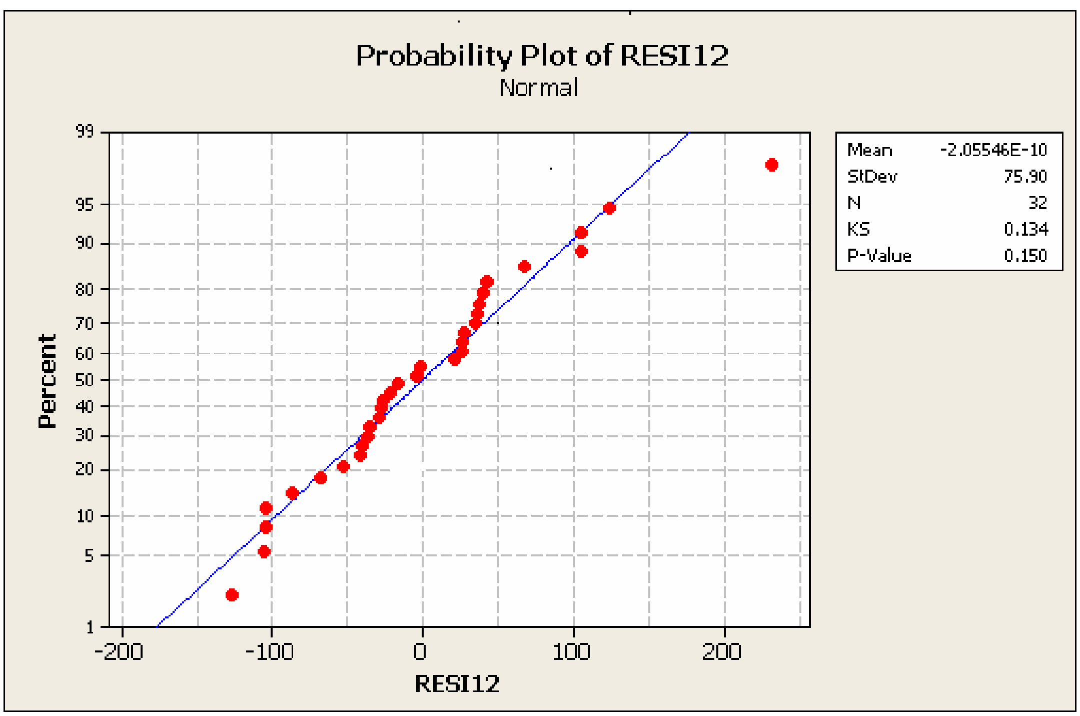
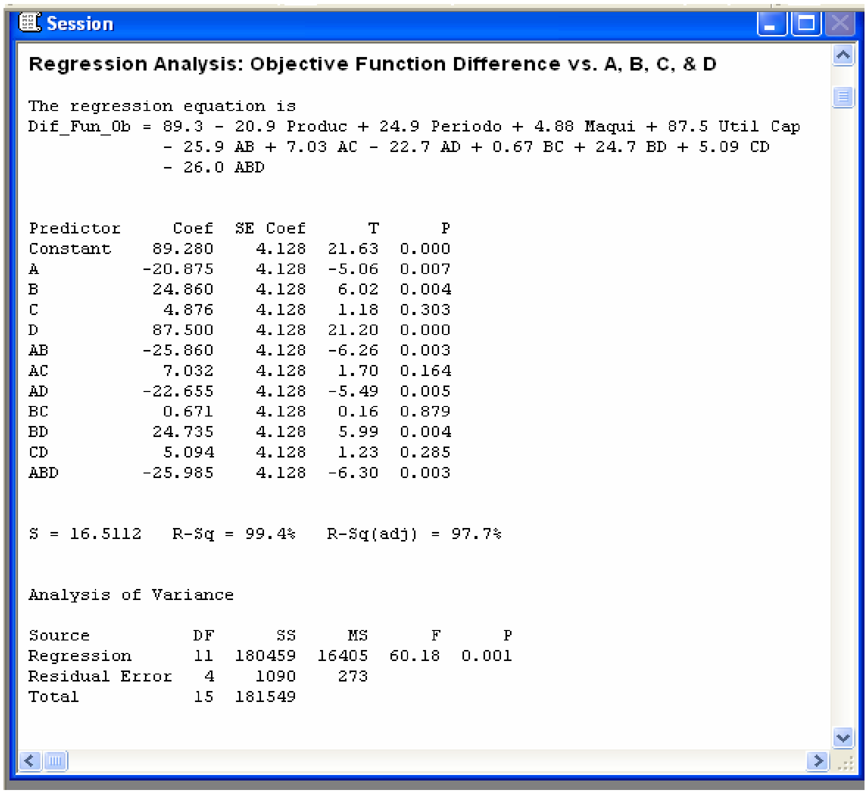


Figure 13: Kolmogorov-Smirnov Test for Normality on the Simplified Model

4.2.2 Analysis of Results Considering Differences

This analysis reduces the size of the data set from 32 to 16 by using the difference of heuristic and algorithm. These differences must be non-negative; the heuristic can perform as good as the algorithm but never better. The statistical analysis cannot consider the full model due to the fact that by reducing the number of data points by half, the full model would render no degrees of freedom. In this experimentation replicates were not an option since the research involves deterministic models, which have no variability in their responses. It was considered appropriate to use the simplified regression model for the analysis with its residual analysis.

The results are presented in Figures 13 and 14. In this case, some main effects (i.e. C) and two-factor interactions (AC, BC and CD) were not significant. Figure 8, which presents results for the simplified model with blocks, shows that even though these effects were significant, the magnitude of the regression coefficients were much smaller compared to other significant effects. These same effects are the ones that are not significant in the results in Figure 12. It is also interesting to notice the regression coefficient for the block effect (in Figure 8) also has a small magnitude, hinting that the discrepancies between the algorithm and the heuristic are not that meaningful. Figure 13 present the residual analysis for the model focused on differences. Residuals look normal and show constant variance. The Kolmogorov-Smirnov test presented in Figure 14 shows the normality null hypothesis cannot be rejected.


Session

Regression Analysis: Objective Function Difference vs. A, B, C, & D

```
The regression equation is
Dif_Fun_Ob = 89.3 - 20.9 Produc + 24.9 Periodo + 4.88 Maqui + 87.5 Util Cap
             - 25.9 AB + 7.03 AC - 22.7 AD + 0.67 BC + 24.7 BD + 5.09 CD
             - 26.0 ABD
```

Predictor	Coef	SE Coef	T	P
Constant	89.280	4.128	21.63	0.000
A	-20.875	4.128	-5.06	0.007
B	24.860	4.128	6.02	0.004
C	4.876	4.128	1.18	0.303
D	87.500	4.128	21.20	0.000
AB	-25.860	4.128	-6.26	0.003
AC	7.032	4.128	1.70	0.164
AD	-22.655	4.128	-5.49	0.005
BC	0.671	4.128	0.16	0.879
BD	24.735	4.128	5.99	0.004
CD	5.094	4.128	1.23	0.285
ABD	-25.985	4.128	-6.30	0.003

S = 16.5112   R-Sq = 99.4%   R-Sq(adj) = 97.7%

Analysis of Variance

Source	DF	SS	MS	F	P
Regression	11	180459	16405	60.18	0.001
Residual Error	4	1090	273		
Total	15	181549			

Figure 14: Regression Analysis for Difference

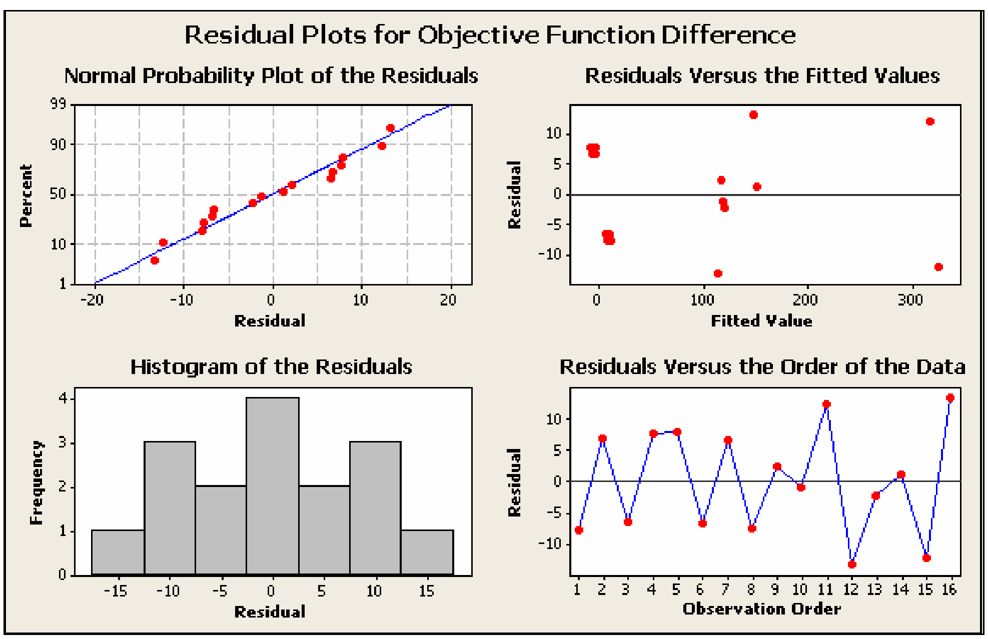


Figure 15: Residual Analysis for Differences

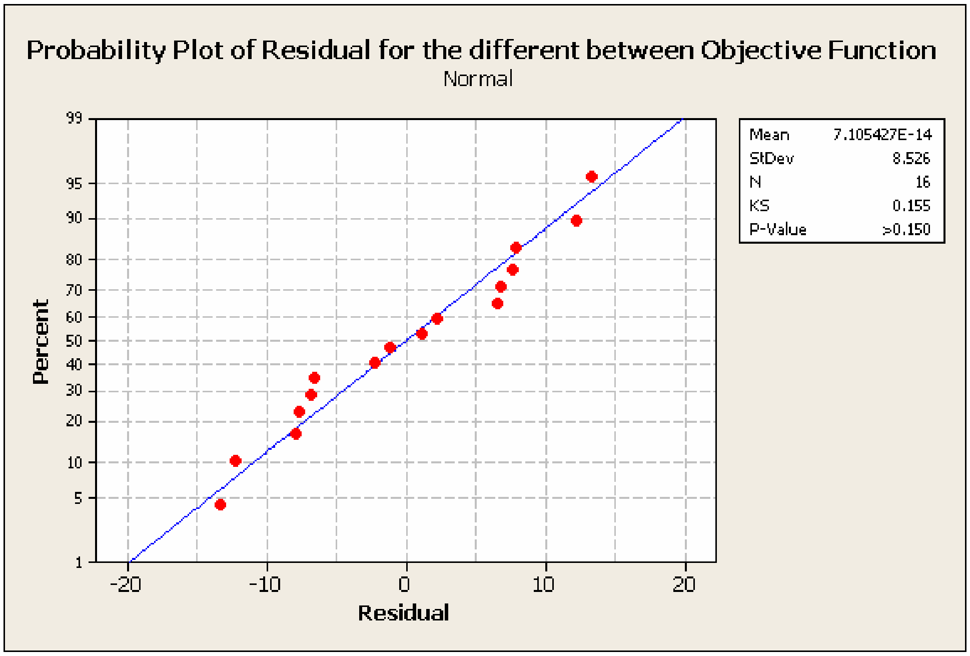


Figure 16: Normal Probability Plot for Residuals of the Difference Model

Figures 16 and 17 present the behavior of the heuristic approach in approximating the optimal value obtained from the algorithm. Figure 16 shows an example in which the heuristic reaches the optimal value in the second run. A worst case scenario is presented in Figure 17, in which the heuristic results fluctuate; however, the magnitude of the observed difference is close to .004 percent. Having this small difference between the values is irrelevant when we compare the computational time savings explained below.

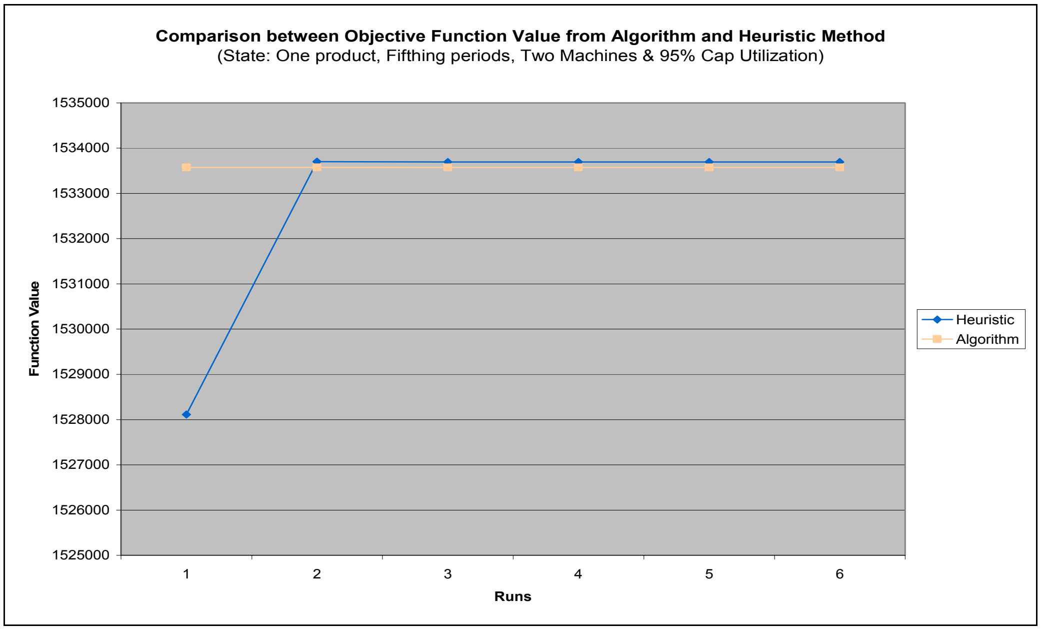


Figure 17: Comparison between Objective Function for Algorithm & Heuristic Model

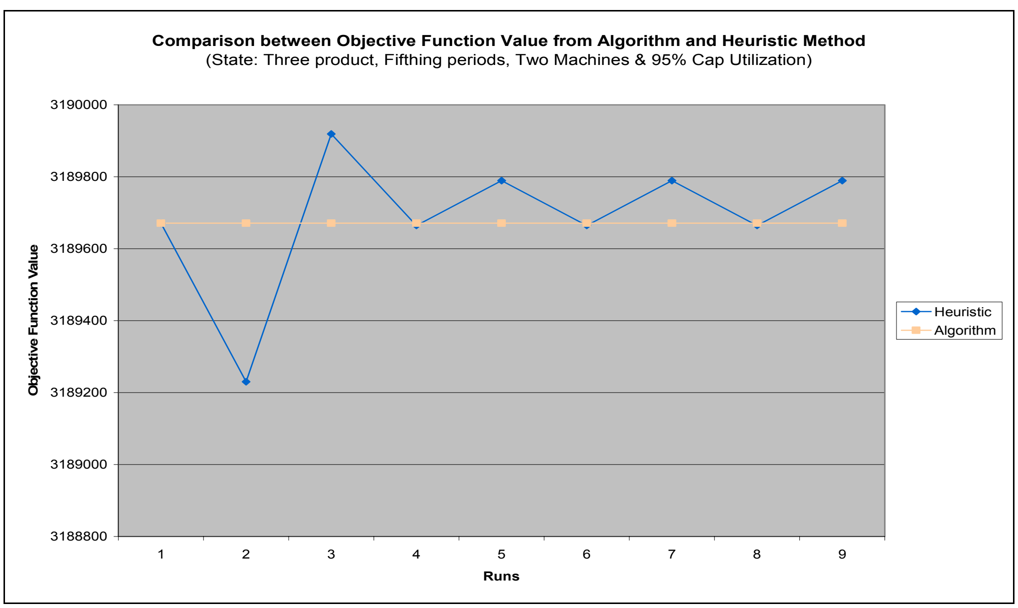


Figure 18: Comparison between objective function from Algorithm & Heuristic Model

Figures 18 and 19 present main effects and interaction graphs. Factor C, which is non-significant, is in the lower left corner of Figure 18. Non-significant interactions are found in Figure 19 in the second column (AC and BC) and in the bottom of the third column (CD).

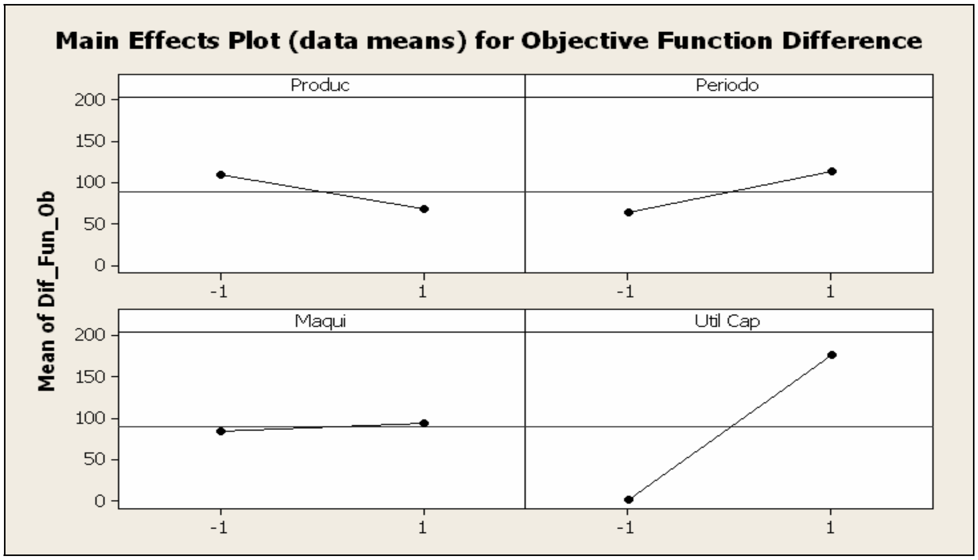


Figure 19: Main Effects for the Difference Model

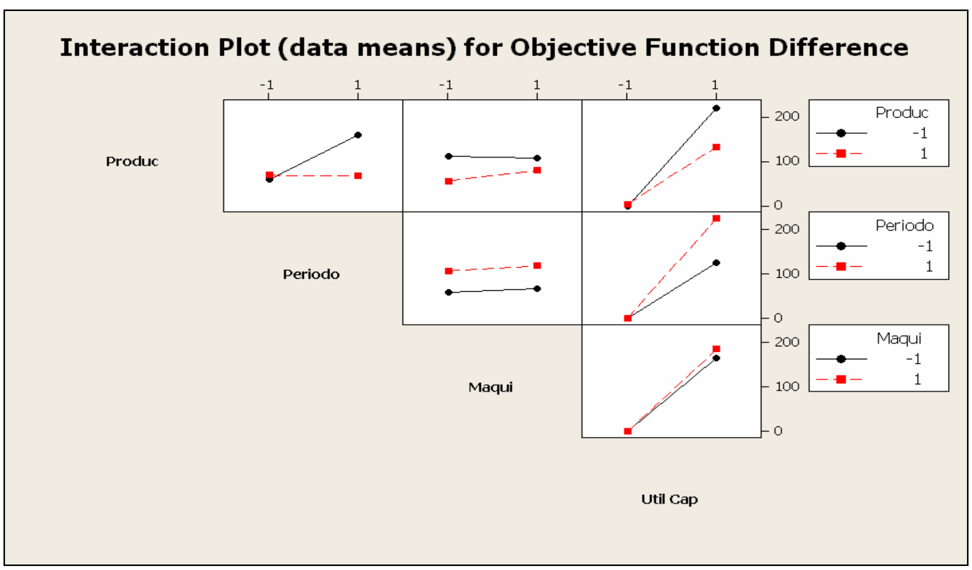


Figure 20: Two-Factor Interactions for the Difference Model

4.2.4 Analysis of Results for Computational Time Requirements

Figure 20 shows the results of a regression analysis for the differences in computational time requirements when moving from the algorithm to the heuristic; the difference is always positive. Main effects C and D and their interaction, CD, were the only significant effects. These two factors are the number of machines (C) and capacity utilization (D). Results show that as machines are added and we increase capacity utilization, the algorithm takes longer to reach the optimum; however the heuristic time requirements remain very low.

The residual analysis, presented in Figure 21, shows difficulty only in upper right graph where fitted and residuals are compared. The right-most point which stands alone relates to the runs for factors at their high values; i.e., three products, 20 periods, three

```
Session

Regression Analysis: Time Difference vs. A, B, C, & D

The regression equation is
Dif Tiempo = 1656 + 778 A + 836 B + 1405 C + 1594 D
             + 313 AB + 589 AC + 716 AD + 741 BC + 822 BD + 1388 CD + 299 ABD


Predictor    Coef  SE Coef     T      P
Constant   1656.5    503.2  3.29  0.030
A           777.5    503.2  1.55  0.197
B           836.0    503.2  1.66  0.172
C          1405.3    503.2  2.79  0.049
D          1594.3    503.2  3.17  0.034
AB          313.3    503.2  0.62  0.567
AC          589.3    503.2  1.17  0.307
AD          715.8    503.2  1.42  0.228
BC          741.2    503.2  1.47  0.215
BD          822.0    503.2  1.63  0.178
CD         1387.7    503.2  2.76  0.051
ABD         299.4    503.2  0.59  0.584


S = 2012.87   R-Sq = 90.8%   R-Sq(adj) = 65.6%


Analysis of Variance

Source          DF         SS        MS     F      P
Regression      11  160293013  14572092  3.60  0.114
Residual Error   4   16206535   4051634
Total           15  176499548
```

Figure 21: Regression Analysis for Difference in Computational Time

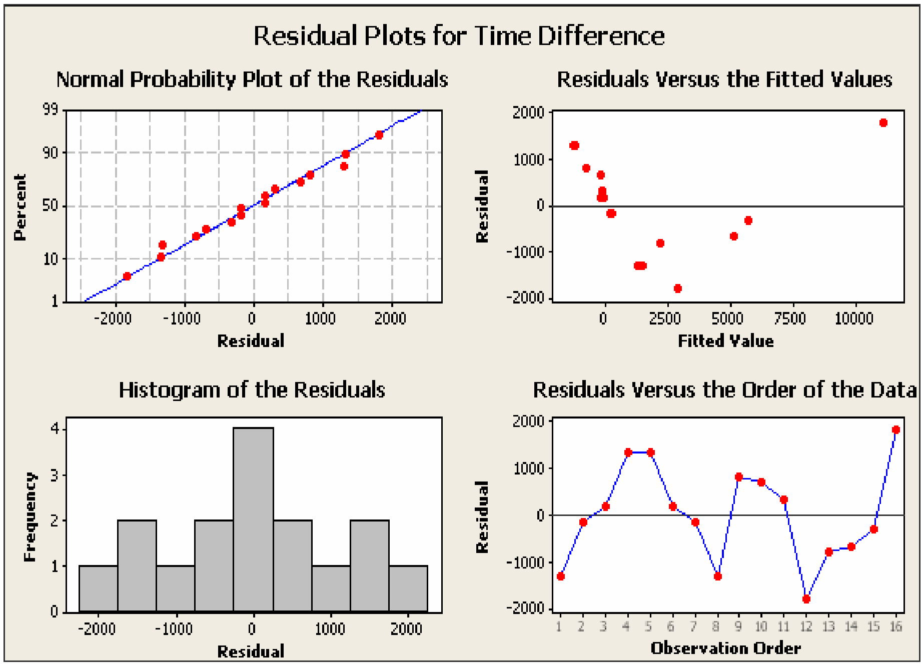


Figure 22: Residual Plot for Difference in Time Requirements

machines, and capacity utilization in 90-95 percent. This run took close to five days for the optimization model to complete, run length which was far away from any other factor combination. Thus, the regression model had difficulty predicting this run length.

Figure 22 presents a box plot which compares algorithm (left) versus the heuristic (right), which speaks loudly of the minimal time requirements of the latter. Given the drastic differences in time requirements and the minimal differences in objective function values, the effectiveness of the heuristic approach cannot be denied! Even for the simplest problem to solve, in which all factors were at the low level, the algorithm takes 35.9 seconds but the heuristic method finds the optimum value after two iterations of 1 second each.

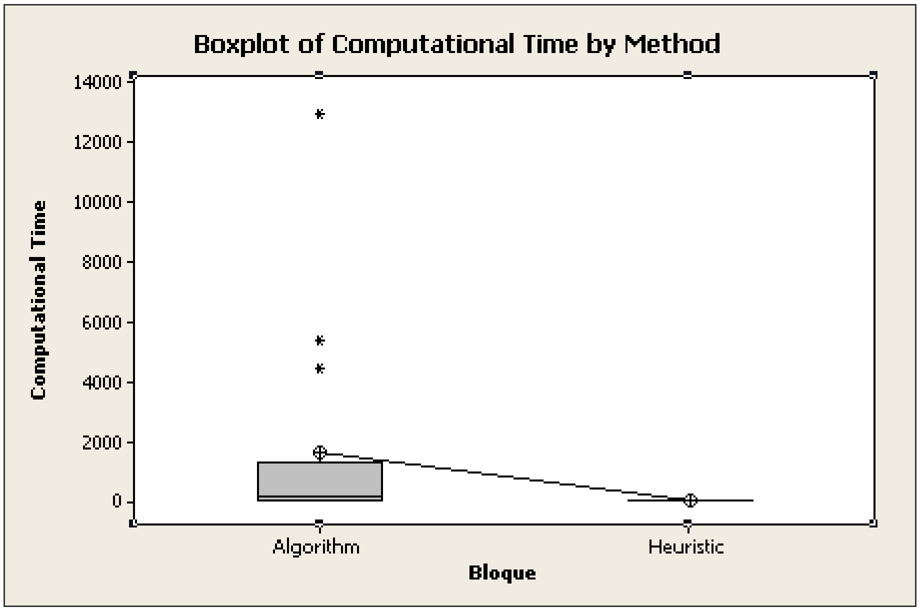


Figure 23: Boxplot of Computational Time Requirements

Chapter 5: Conclusions and Future Research

The heuristic method developed as part of this research complies with our expectation by solving a set of problems defined by the experimental design requiring minimal execution time with excellent proximity to the optimum solutions provided by the algorithm.

The differences in objective function values between heuristic and algorithm results were always below .02 percent. The time requirements for the heuristic were always minimal. On the computational time differences, we can appreciated at the exercise with all factors on high level the extreme difference on computational time. The optimization algorithm gives us a solution after 9 days and twenty hours while the heuristic method gives us a solution after 0.12 second. The main issues demanding additional execution time for the algorithm were the number of machines (factor C) and capacity utilization (factor D). It was noticed that if capacity utilization was low, the algorithm would reach optimality quickly.

The work performed by Trigeiro (1989) and Nabil (2002) served as a starting point for creating our own production scheduling scenarios. The problems they proposed in 1989 and 2002, respectively, did not consider bills of materials, allowed for only one machine, and did not force the algorithm to run for a significant amount of time to allow an adequate statistical analysis. The development of the flashlight case study give us the opportunity of analyze the heuristic method and change the levels of factor to force the method to give us a solution under complex conditions.

The optimization tools were developed in Visual Basic for Applications (VBA), a Visual Basic version which resides in the Microsoft Office software family. Specifically, the VBA code was developed within Excel. This software was designed to create the objective function and constraints for the optimization model, complying with a specific linear programming model structure known as LP format. The software then invokes an optimization problem solver, after which results are then printed into a user friendly format. The programming of the optimization algorithm serves it purpose on the research because it help us to compare to solution obtained by the heuristic method and to assess the closeness to the optimal solutions.

This work can be expanded for future research in various ways:

i – Considering other product scenarios with the needed capacity and set-up considerations. The point is that a larger case study menu is a “nice to have” for the community of Management Science researchers interested in production planning and scheduling solution approaches.

ii - Increasing the current (flashlight) case study by including more machines, products, periods and larger use of capacity utilization. The emphasis could move to “beyond full capacity” and the competition between overtime and backlog costs. Increasing the problem size is not a trivial task since some new factor combinations might require an excessive amount of time or might require a more powerful computer for reaching the optimum.

iii – Comparing the traditional simplex algorithm as optimization engine against newer approaches such as genetic algorithm, simulated annealing, tabu search, ants colonies, and genetic algorithm. The interest of this activity would be on the effectiveness of these newer approaches against the standard or traditional (simplex) in solving available case studies related to production planning and scheduling scenarios.

Bibliography

Articles:

[1] Absi (2005). Multi-item capacitated lot-sizing problem with setup times and shortage costs: Polyhedral results. Technical Report LIP6 2005/009.

[2] Barker J.R. (1994).MRP Π, Real time Scheduling and SBM. Assembly Automation Journal; Volume 14. Issue 2. Page 22-28.

[3] Baker, K. R. (1999). Heuristic Procedures for scheduling Job Families with Setups and Due Dates. Naval Research Logistics, Volume 46 Page: 978-991.

[4] Boubekri (2001) Technology enablers for supply chain management. Integrated Manufacturing System. Volume: 12 Issue: 6 Page 394-399

[5] Carr, S.A. (1993) Exact Analysis of B/C Stock Policy, Tech. Report 1051, School of OR&IE, Cornell University, Ithaca, NY.

[6] Chang, Hasting and White (1994). A Very Fast Production Scheduler. International Journal of Operations & Production Management. Volume 14. Issue 8. Page: 88-101.

[7] Chen and Wang (1997). A linear programming model for integrated steel production and distribution planning International Journal of Operations & Production Management; Volume: 17 Issue: 6; Case study. Page 592-610

[8] Chung and Snyder, (2000) ERP adoption: a technological evolution approach. International Journal of Agile Management Systems Volume: 2 Issue: 1 Page 24-32.

[9] Gaglioppa, Miller and Benjaafar (2004) Multi-Task! Multi-Stage Production Planning and Scheduling for Process Industries.

[10] Koclar and Soral (2004). Development of a Production Planning Model for Process Industry Environment. Page: 15-18.

[11] Lau, Zhao and Lai., (2002). Survey of MRP Π Implementation and Benefits in Mainland China and Hong Kong.Production and Inventory Management Journal. Third/Fourth Quarter. Page: 65-71

[12] Metaxiotis and Psarras, (2004). The contribution of neural networks and genetic algorithms to business decision support. Management Decision. Volume: 42 Issue: 2 Page 229-242

[13] Mingyun Chen (2001). A model for integrated production planning in cellular systems. Integrated Manufacturing Systems. Volume 12. Issue 4. Page 275-284.

[14] Muckstadt, Murray and Rappold (2001). Capacitated Production Planning and Inventory Control when Demand is Unpredictable for Most Items: The No BIC Strategy. Page: 1-36.

[15] Porter, Little, Laakmann and Schotten (1996). .Production planning and control system developments in Germany: International Journal of Operations & Production Management; Volume: 16 Issue: 1; Page: 27-39.

[16] Qui, Petterson and Cao. (2002). Scheduling Machines Using A Constraints Management Heuristic. Production and Inventory Management Journal. FisrtlSecond Quarter. Page: 35-43

[17] Santos, Powell and Sarshar, (2002) Evolution of management theory: the case of production management in construction. Management Decision. Volume: 40 Issue: 8 Page 788-796

[18] Seyed-Mahmoud Aghazadeh., (2003) MRP contributes to a company's profitability. Journal Assembly Automation. Volume: 23 Issue: 3 Page: 257 - 265.

[19] W. Trigeiro, L.J. Thomas and J.O. McClain, Capacitated lot sizing with set-up times. Management Sci. 35 3 (1989), pp. 353–366.

[20] W.H. Lp (1998). Manufacturing integration strategy using MRP Π and RTMs: a case study in South China. Integrated Manufacturing Systems; Volume: 9 Issue: 1; Case study. Page 41-49.

Books:

[21] Goldratt, E.M.(1992) The Goal, 2nd rev. ed. Croton-on-Hudson, N.Y.: North River Press.

[22] Peterson, Silver, Wiley & Sons.(1979) Material Requirement Planning. Decision System for Inventory Management and Production Planning. Page 459-474

[23] Silver, Pyke, Perterson.(1998) Inventory Management and Production Planning and Scheduling, 3rd ed.,

[24] Taha. H. A. (2003). Operations Research: An Introduction (7th Edition).

Page: 443-461

[25] Voss, Stefan (2003). Introduction to Computational Optimization Models for Production Planning in a Supply Chain. Chapter 1-5

[26] Winston, W. L. (1994). Recent Developments in Inventory Theory. Operations Research Applications and Algorithms, Third Edition.

Page: 940-960.

Appendix A. Visual Basic Code for Heuristic Model

```
Dim Horizon, hz As Integer
Dim tmenoslt, tmenos1 As Variant
Dim col, row As Integer
Dim NumAssemblies As Integer
Dim NumResources As Integer
Dim maxchar As Integer
Dim totdem As Double
Dim totbom As Long
Dim LargeM As Long

Dim FirstRun, fila, prox, addi, FilenameLP, HeuristicOrAlgorithm As String

Dim Assembly(), thisWB As String
Dim initialinventory() As Double
Dim yield() As Double
Dim initialbacklog() As Double
Dim finalinventory() As Double
Dim CumDemand() As Double
Dim CumProd() As Double
Dim invcost() As Double
Dim backlogcost() As Double
Dim OTCost() As Double
Dim lt() As Integer
Dim LotSize() As Integer
Dim resource() As String

Dim cAssIdx As New Collection
Dim cResIdx As New Collection
Dim cMatReq As New Collection
Dim cDemand As New Collection
Dim cRequired As New Collection
Dim cCapacity As New Collection
Dim cMaxOT As New Collection
Dim cInProcess As New Collection
Dim cUsers() As New Collection
Dim cUsedBy() As New Collection
Dim cProdSched As New Collection
Dim cInvSched As New Collection
Dim cBackLog As New Collection
Dim cSetup As New Collection
Dim cUsersOfResource() As New Collection
Sub lpformat()
   readdata
   createfile
```

```
    objective
    startedproduction
    initialinv
    finalinv
    initialbl

    matbal

    MsgBox ("HeuristicOrAlgorithm = " & HeuristicOrAlgorithm)

    If HeuristicOrAlgorithm = "a" Then
        ProductionYesOrNo
        LotSizeRequired
        deltagamma
        OneGamma
    End If

    resources
    maxovertime

    ende

    If HeuristicOrAlgorithm = "a" Then
        bounds
    End If

    enddata
    'retval = Shell("c:\EatonScheduling\lp.exe", 1)
    'ProdSchedule
    End
End Sub
Sub createfile()
    Set fs = CreateObject("Scripting.FileSystemObject")
    fs.CreateTextFile FilenameLP
End Sub

Sub checklength(addi)
    If Len(addi) + Len(prox) >= maxchar Then
        addi = (Chr(13) + Chr(10)) & addi
        prox = ""
    End If
    '==========================================
    'prox = proximo pedazo de 250 caracteres que
    'le estoy pegando a fila.
    'prox y el pedazo
    'de fila que viene despues del ultimo vbCrLf
    'son identicos.
    'addi es el proximo pedazo que le estoy pegando
```

```
    'a prox y quiero ver si pasaria a prox de los
    '250 caracteres.
    'Si lo pasaria entonces no se lo anado sino
    'que empiezo otro prox poniendo (Chr(13) + Chr(10)) &
    'antes de addi y hago prox = ""
    '==============================================
    prox = prox + addi
    fila = fila & addi
End Sub
Sub writetotxtfile(line)
    Set fs = CreateObject("Scripting.FileSystemObject")
    Set f = fs.GetFile(FilenameLP)
    Set a = f.OpenAsTextStream(8, -2)
    a.WriteLine line
    a.Close
    fila = ""
    prox = ""
End Sub
Sub readdata()
    Workbooks(ActiveWorkbook.Name).Worksheets("parameters").Activate
    FilenameLP = Cells(2, 1)
    maxchar = Cells(4, 1)
    Horizon = Cells(6, 1)
    hz = Horizon

    HeuristicOrAlgorithm = Cells(10, 1)
    'MsgBox ("HeuristicOrAlgorithm at readdata = " & HeuristicOrAlgorithm)
    If HeuristicOrAlgorithm = "h" Then
        FirstRun = Cells(12, 1)
        If FirstRun <> "y" And FirstRun <> "n" Then
            FirstRun = InputBox("Is this the first of a sequence of heuristic runs (y or n)")
        End If
        Cells(12, 1) = FirstRun
        If FirstRun = "" Then
            End
        End If
        If FirstRun = "y" Then
            Call InitializeCapacity
        End If
    ElseIf HeuristicOrAlgorithm = "a" Then
        Call InitializeCapacity
        Cells(12, 1) = "Running Algorithm"
    End If

    Workbooks(ActiveWorkbook.Name).Worksheets("ProductData").Activate
    NumAssemblies = 0
    row = 2
    Do Until Cells(row, 1) = ""
```

```
        NumAssemblies = NumAssemblies + 1
        ReDim Preserve Assembly(NumAssemblies + 1)
        ReDim Preserve initialinventory(NumAssemblies + 1)
        ReDim Preserve yield(NumAssemblies + 1)
        ReDim Preserve initialbacklog(NumAssemblies + 1)
        ReDim Preserve finalinventory(NumAssemblies + 1)
        ReDim Preserve lt(NumAssemblies + 1)
        ReDim Preserve invcost(NumAssemblies + 1)
        ReDim Preserve backlogcost(NumAssemblies + 1)
        ReDim Preserve LotSize(NumAssemblies + 1)
        ReDim Preserve CumDemand(NumAssemblies + 1)
        ReDim Preserve CumProd(NumAssemblies + 1)
        CumDemand(NumAssemblies) = 0
        CumProd(NumAssemblies) = 0
        Assembly(NumAssemblies) = Cells(row, 1)
        cAssIdx.Add Item:=NumAssemblies, key:=Assembly(NumAssemblies)
        'If Assembly(NumAssemblies) = "1B85620G03X" Then
            'MsgBox (cAssIdx("1B85620G03X"))
        'End If
        initialinventory(NumAssemblies) = Cells(row, 2)
        finalinventory(NumAssemblies) = Cells(row, 3)
        lt(NumAssemblies) = Cells(row, 4)
        LotSize(NumAssemblies) = Cells(row, 5)
        invcost(NumAssemblies) = Cells(row, 6)
        backlogcost(NumAssemblies) = Cells(row, 7)
        initialbacklog(NumAssemblies) = Cells(row, 8)
        yield(NumAssemblies) = Cells(row, 9)
        row = row + 1
    Loop
    ReDim cUsers(NumAssemblies)
    ReDim cUsedBy(NumAssemblies)

    '=============================================================
    '============Read MatReq Related Data=====================
    '=============================================================
    Workbooks(ActiveWorkbook.Name).Worksheets("WhereUsed").Activate
    row = 2
    totbom = 0
    Do Until Cells(row, 1) = ""
        cMatReq.Add Item:=Cells(row, 3), key:=Cells(row, 1) & Cells(row, 2)
        i = cAssIdx(Cells(row, 1))
        cUsers(i).Add Item:=Cells(row, 2)
        j = cAssIdx(Cells(row, 2))
        cUsedBy(j).Add Item:=Cells(row, 1)
        totbom = totbom + Cells(row, 3)
        row = row + 1
    Loop
'=================================================================
```

```
'===================Read Demand Related Data====================
'=============================================================
    'Workbooks(ActiveWorkbook.Name).Worksheets("Demand").Activate
    Workbooks(ActiveWorkbook.Name).Worksheets("ProductData").Activate
    row = 2
    totdem = 0
    Do Until Cells(row, 2) = ""
        col = 11
        t = 1
        Do Until Cells(row, col) = ""
            'cDemand.Add Item:=Cells(row, col), key:=(Cells(1, col) & Cells(row, 1))
            cDemand.Add Item:=Cells(row, col), key:=(Cells(row, 1) & Cells(1, col))
            i = cAssIdx(Cells(row, 1))
            CumDemand(i) = CumDemand(i) + Cells(row, col)
            totdem = totdem + Cells(row, col)
            col = col + 1
        Loop
        row = row + 1
    Loop
    LargeM = (totdem * totbom) + 10000
'=============================================================
'===================Read Resources Related Data=================
'=============================================================
    Workbooks(ActiveWorkbook.Name).Worksheets("Resources").Activate
    row = 2
    NumResources = 0
    Do Until Cells(row, 1) = ""
        NumResources = NumResources + 1
        ReDim Preserve resource(NumResources)
        ReDim Preserve OTCost(NumResources)
        ReDim Preserve cUsersOfResource(NumResources)
        resource(NumResources) = Cells(row, 1)
        cResIdx.Add Item:=NumResources, key:=resource(NumResources)
        OTCost(NumResources) = Cells(row, 2)
        row = row + 1
    Loop
    row = 2
    Do Until Cells(row, 3) = ""
        cCapacity.Add Item:=Cells(row, 5), _
        key:=Cells(row, 3) & Cells(row, 4)
        row = row + 1
    Loop
    row = 2
    Do Until Cells(row, 3) = ""
        cMaxOT.Add Item:=Cells(row, 6), _
        key:=Cells(row, 3) & Cells(row, 4)
        row = row + 1
    Loop
```

```
    row = 2
    Do Until Cells(row, 7) = ""
        cRequired.Add Item:=Cells(row, 9), _
        key:=Cells(row, 7) & Cells(row, 8)
        cSetup.Add Item:=Cells(row, 10), _
        key:=Cells(row, 7) & Cells(row, 8)
        ir = cResIdx(Cells(row, 7))
        cUsersOfResource(ir).Add Item:=Cells(row, 8)

        row = row + 1
    Loop

'=================================================================
'============Read InProcessProduction Related Data=============
'=================================================================
    Workbooks(ActiveWorkbook.Name).Worksheets("InProcess").Activate
    r = 2
    Do Until Cells(r, 1) = ""
        cInProcess.Add Item:=Cells(r, 3), key:=(Cells(r, 1) & Cells(r, 2))
        r = r + 1
    Loop

End Sub
Sub objective()
    'writetotxtfile ("#")
    checklength ("MIN: ")
    For s = 1 To NumAssemblies
        For t = 1 To Horizon - lt(s)
            If t < 10 Then t = 0 & t
            'checklength ("+" & (Horizon - t) & "*x_" & Assembly(s) & "_" & t)
        Next
    Next
    For s = 1 To NumAssemblies
        For t = 1 To Horizon
            If t < 10 Then t = 0 & t
            checklength ("+" & invcost(s) & "*I_" & Assembly(s) & "_" & t)
            checklength ("+" & backlogcost(s) & "*B_" & Assembly(s) & "_" & t)
        Next
    Next
    'For s = 1 To NumAssemblies
        'tini = 1
        'If lt(s) > 0 And cInProcess(Assembly(s) & "00") > 0 Then tini = 0
        'For t = tini To Horizon - lt(s)
            'If t < 10 Then t = 0 & t
            'checklength ("+" & "0.00001" & "*Delta_" & Assembly(s) & "_" & t)
        'Next
    'Next
    For k = 1 To NumResources
```

```
        For t = 1 To Horizon
            If t < 10 Then t = 0 & t
            checklength ("+" & OTCost(k) & "*O_" & resource(k) & "_" & t)
        Next
    Next
    checklength (";")
    writetotxtfile (fila)
End Sub
Sub initialinv()
    writetotxtfile ("#")
    For s = 1 To NumAssemblies
        t = 0 & 0
        checklength ("II_" & Assembly(s) & "_" & t & ": ")
        checklength ("I_" & Assembly(s) & "_" & t & "=" & _
        initialinventory(s) & ";")
        writetotxtfile (fila)
    Next
End Sub
Sub finalinv()
    writetotxtfile ("#")
    For s = 1 To NumAssemblies
        t = Horizon
        If t < 10 Then t = 0 & t
        checklength ("FI_" & Assembly(s) & "_" & t & ": ")
        checklength ("I_" & Assembly(s) & "_" & t & ">" & _
        finalinventory(s) & ";")
        writetotxtfile (fila)
    Next
End Sub
Sub initialbl()
    writetotxtfile ("#")
    For s = 1 To NumAssemblies
        t = 0 & 0
        checklength ("IBL_" & Assembly(s) & "_" & t & ": ")
        checklength ("B_" & Assembly(s) & "_" & t & "=" & _
        initialbacklog(s) & ";")
        writetotxtfile (fila)
    Next
End Sub
Sub startedproduction()
    'Started_lq8811_n2: x_lq8811_n2=0;
    'Started_lq8811_n1: x_lq8811_n1=0;
    'Started_lq8811_00: x_lq8811_00=0;
    writetotxtfile ("#")
    For s = 1 To NumAssemblies
        For t = 1 - lt(s) To 0
            If t < 0 Then
                t1 = "n" & Abs(t)
```

```
        ElseIf t < 10 Then
            t1 = 0 & t
        End If
        checklength ("Started_" & Assembly(s) & "_" & t1 & ": ")
        checklength ( _
        "x_" & Assembly(s) & "_" & t1 & "=" & cInProcess(Assembly(s) & t1) & ";")
        writetotxtfile (fila)
    Next t
    If s < NumAssemblies Then writetotxtfile ("#")
Next s

End Sub
Sub matbal()
'MatBal_aj8172_01: +I_aj8172_00+x_aj8172_n1-D_aj8172_01-I_aj8172_01=0;
    writetotxtfile ("#")
    For s = 1 To NumAssemblies
        For t = 1 To Horizon
            tmenoslt = t - lt(s)
            If tmenoslt < 0 Then
                tmenoslt = "n" & Abs(tmenoslt)
            ElseIf tmenoslt < 10 Then
                tmenoslt = 0 & tmenoslt
            End If
            If t < 10 Then t = 0 & t
            tmenos1 = t - 1
            If tmenos1 < 10 Then tmenos1 = "0" & tmenos1
            checklength ("MatBal_" & Assembly(s) & "_" & t & ": ")
            checklength ( _
            "+I_" & Assembly(s) & "_" & tmenos1 & _
            "+" & yield(s) & "*x_" & Assembly(s) & "_" & tmenoslt)

            For Each u In cUsers(s)
                uid = cAssIdx(u)
                mr = cMatReq(Assembly(s) & Assembly(uid))
                If CInt(t) <= Horizon - lt(uid) And _
                cMatReq(Assembly(s) & Assembly(uid)) > 0 Then
                    checklength ("-" & cMatReq(Assembly(s) & Assembly(uid)) & _
                    "*x_" & Assembly(uid) & "_" & t)
                End If
            Next

            'checklength ( _
            "-I_" & Assembly(s) & "_" & t & "+B_" & Assembly(s) & "_" & t)

            If CumDemand(s) > 0 Then
                checklength ("+B_" & Assembly(s) & "_" & t)
                checklength ("-B_" & Assembly(s) & "_" & tmenos1)
            End If
```

```
            checklength ( _
            "-I_" & Assembly(s) & "_" & t)

            checklength ("=" & cDemand(Assembly(s) & t) & ";")
            writetotxtfile (fila)
        Next
        If s < NumAssemblies Then writetotxtfile ("#")
    Next
End Sub
'ProdYN_aj8172_01: Delta_aj8172_01*8000-x_aj8172_01>0;
Sub ProductionYesOrNo()
    writetotxtfile ("#")
    For s = 1 To NumAssemblies
        tini = 1
        If lt(s) > 0 And cInProcess(Assembly(s) & "00") > 0 Then tini = 0
        For t = tini To Horizon - lt(s)
            If t < 0 Then
                t1 = "n" & Abs(t)
            ElseIf t < 10 Then
                t1 = 0 & t
            Else
                t1 = t
            End If
            checklength ("MadeYorN_" & Assembly(s) & "_" & t1 & ": ")
            checklength ( _
            LargeM & "*Delta_" & Assembly(s) & "_" & t1 & _
            "-x_" & Assembly(s) & "_" & t1 & ">0;")
            writetotxtfile (fila)
        Next t
        If s < NumAssemblies Then writetotxtfile ("#")
    Next s
End Sub
'LotSizeReq_aj8172_01: -Delta_aj8172_01*100+x_aj8172_01>0;
Sub LotSizeRequired()
    writetotxtfile ("#")
    For s = 1 To NumAssemblies
        For t = 1 To Horizon - lt(s)
            t1 = t
            If t < 0 Then
                t1 = "n" & Abs(t)
            ElseIf t < 10 Then
                t1 = 0 & t
            End If
            checklength ("LotSize_" & Assembly(s) & "_" & t1 & ": ")
            checklength ( _
            "-" & LotSize(s) & "*Delta_" & Assembly(s) & "_" & t1 & _
            "+" & "x_" & Assembly(s) & "_" & t1 & ">0;")
```

```
            writetotxtfile (fila)
        Next t
        If s < NumAssemblies Then writetotxtfile ("#")
    Next s
End Sub
Sub resources()
    writetotxtfile ("#")
    For t = 1 To Horizon
        For k = 1 To NumResources
            t1 = t
            If t < 10 Then
                t1 = 0 & t
            End If
            checklength ("Capacity_" & resource(k) & "_" & t1 & ": ")
            For Each ensamblaje In cUsersOfResource(k)
            s = cAssIdx(ensamblaje)
            'For s = 1 To NumAssemblies
                If t <= Horizon - lt(s) And _
                cRequired(resource(k) & ensamblaje) > 0 Then
                    checklength ( _
                    "+" & cRequired(resource(k) & Assembly(s)) & _
                    "*x_" & Assembly(s) & "_" & t1)
                End If
            Next
            If HeuristicOrAlgorithm = "a" Then
            For Each prod In cUsersOfResource(k)
                s = cAssIdx(prod)
                If t <= Horizon - lt(s) And _
                cSetup(resource(k) & Assembly(s)) > 0 And _
                cRequired(resource(k) & Assembly(s)) > 0 Then
                    checklength ( _
                    "+" & cSetup(resource(k) & Assembly(s)) & _
                    "*Delta_" & Assembly(s) & "_" & t1)
                    If t > 1 Then
                        checklength ( _
                        "-" & cSetup(resource(k) & Assembly(s)) & _
                        "*Gamma_" & Assembly(s) & "_" & resource(k) & "_" & t1)
                    ElseIf t = 1 And cInProcess(Assembly(s) & "00") > 0 Then
                        checklength ( _
                        "-" & cSetup(resource(k) & Assembly(s)) & _
                        "*Gamma_" & Assembly(s) & "_" & resource(k) & "_" & t1)
                    End If
                End If
            Next
            End If
            checklength ("+E_" & resource(k) & "_" & t1 & "-O_" & resource(k) & "_" & t1)
            checklength ("=" & cCapacity(resource(k) & t1) & ";")
            writetotxtfile (fila)
```

```
        Next
    Next
End Sub
Sub deltagamma()
    writetotxtfile ("#")
    'For p = 1 To NumAssemblies
        'For k = 1 To NumResources
        'If cRequired(resource(k) & Assembly(p)) > 0 Then
        For k = 1 To NumResources
         For Each prod In cUsersOfResource(k)
         p = cAssIdx(prod)
            t1 = 2
            If lt(p) > 0 And cInProcess(Assembly(p) & "00") > 0 Then t1 = 1
            For t = t1 To Horizon - lt(p)
                If t < 10 Then t = 0 & t
                tm1 = t - 1
                If tm1 < 10 Then tm1 = 0 & tm1
                checklength ("DG_" & Assembly(p) & "_" & resource(k) & "_" & t & ": ")
                checklength ( _
                "Delta_" & Assembly(p) & "_" & t & _
                "+Delta_" & Assembly(p) & "_" & tm1 & _
                "-2*Gamma_" & Assembly(p) & "_" & resource(k) & "_" & t)
                checklength (">0;")
                writetotxtfile (fila)
            Next
        Next
    Next
    writetotxtfile ("#")
    'For p = 1 To NumAssemblies
        'For k = 1 To NumResources
        'If cRequired(resource(k) & Assembly(p)) > 0 Then
        For k = 1 To NumResources
         For Each prod In cUsersOfResource(k)
         p = cAssIdx(prod)
            tini = 2
            If lt(p) > 0 And cInProcess(Assembly(p) & "00") > 0 Then tini = 1
            For t = tini To Horizon - lt(p)
                If t < 10 Then t = 0 & t
                checklength ("rG_" & Assembly(p) & "_" & resource(k) & "_" & t & ": ")
                checklength ("Gamma_" & Assembly(p) & "_" & resource(k) & "_" & t & _
                "<" & LargeM * cRequired(resource(k) & Assembly(p)) & ";")
                writetotxtfile (fila)
            Next
        Next
    Next
End Sub

Sub maxovertime()
```

```
    writetotxtfile ("#")
    For t = 1 To Horizon
        For k = 1 To NumResources
            t1 = t
            If t < 10 Then
                t1 = 0 & t
            End If
            checklength ("MaxOT_" & resource(k) & "_" & t1 & ": ")
            checklength ("O_" & resource(k) & "_" & t1)
            checklength ("<" & cMaxOT(resource(k) & t1) & ";")
            writetotxtfile (fila)
        Next
    Next
End Sub
Sub OneGamma()
    writetotxtfile ("#")
    For t = 1 To Horizon
        npr = 0
        For p = 1 To NumAssemblies
            If t <= Horizon - lt(p) Then
                If t > 1 Then
                    npr = npr + 1
                ElseIf t = 1 And cInProcess(Assembly(p) & "00") > 0 Then
                    npr = npr + 1
                End If
            End If
        Next
        If npr > 0 Then
            For k = 1 To NumResources
                t1 = t
                If t < 10 Then
                    t1 = 0 & t
                End If
                checklength ("OneGamma_" & resource(k) & "_" & t1 & ": ")
                'For p = 1 To NumAssemblies
                For Each prod In cUsersOfResource(k)
                    p = cAssIdx(prod)
                    If t <= Horizon - lt(p) And _
                    cRequired(resource(k) & Assembly(p)) > 0 Then
                        If t >= 2 Then
                            checklength ("+Gamma_" & Assembly(p) & "_" & resource(k) & "_" & t1)
                        ElseIf t = 1 And cInProcess(Assembly(p) & "00") > 0 Then
                            checklength ("+Gamma_" & Assembly(p) & "_" & resource(k) & "_" & t1)
                        End If
                    End If
                Next
                checklength ("<1;")
                writetotxtfile (fila)
```

```
      Next
    End If
  Next
End Sub
Sub ende()
  writetotxtfile (":ENDE")
End Sub
Sub bounds()
'BV BOUND     Delta_aj8172_01
writetotxtfile ("BOUNDS")
For s = 1 To NumAssemblies
  t1 = 1
  If lt(s) > 0 And cInProcess(Assembly(s) & "00") > 0 Then t1 = 0
  For t = t1 To Horizon - lt(s)
    If t < 10 Then t = 0 & t
    checklength (" BV  BOUND" & "     Delta_" & Assembly(s) & "_" & t)
    writetotxtfile (fila)
  Next
Next
'For s = 1 To NumAssemblies
For k = 1 To NumResources
  For Each prod In cUsersOfResource(k)
    s = cAssIdx(prod)
  'If cRequired(resource(k) & Assembly(s)) > 0 Then
  tini = 2
  If lt(s) > 0 And cInProcess(Assembly(s) & "00") > 0 Then tini = 1
  For t = tini To Horizon - lt(s)
    If t < 10 Then t = 0 & t
    checklength (" BV  BOUND" & _
    "     Gamma_" & Assembly(s) & "_" & resource(k) & "_" & t)
    writetotxtfile (fila)
  Next
  'End If
  Next
Next
End Sub
Sub enddata()
  writetotxtfile ("ENDATA")
End Sub
Sub ProdSchedule()
  readdata
  hz = Horizon
  ans = MsgBox("ProdSchedule", vbYesNo)
  If ans = vbNo Then End
  InputDataFromTextFile

  Workbooks(ActiveWorkbook.Name). _
  Worksheets("ProdSchedule").Activate
```

```
Sheets("ProdSchedule").Select
Cells.Select
Selection.Delete
Range("A1").Select

fstrowx = fstrow("x_")
lstrowx = lstrow("x_")
fstrowi = fstrow("I_")
lstrowi = lstrow("I_")
fstrowbl = fstrow("B_")
lstrowbl = lstrow("B_")
fstrowot = fstrow("O_")
lstrowot = lstrow("O_")
fstrowe = fstrow("E_")
lstrowe = lstrow("E_")

Workbooks(ActiveWorkbook.Name). _
Worksheets("ProdSchedule").Activate
Sheets("ProdSchedule").Select
Cells(1, 1) = "ProdSched"
For t = 1 To hz
   If t < 10 Then t = 0 & t
   Cells(t + 1, 1) = t
Next
For t = 0 To hz
   If t < 10 Then t = 0 & t
   Cells(hz + 2 + t + 2, 1) = t
Next
For t = 1 To hz
   If t < 10 Then t = 0 & t
   Cells(2 * (hz + 2) + 1 + t, 1) = t
Next
Cells(2 * (hz + 2) + 1, 1) = "ExcessCap"

Workbooks(ActiveWorkbook.Name).Worksheets("LPResults").Activate
maxc = 0

row = fstrowx
Debug.Print "x_"; Now; row
Do Until row > lstrowx
      Var = Worksheets("LPResults").Cells(row, 1)
      howmany = Worksheets("LPResults").Cells(row, 2)
      If Var = "x_1B85620G03X_00" Then
         MsgBox (BkVar(Var, 1))
         MsgBox (cAssIdx("1B85620G03X"))
      End If
      ii = cAssIdx(BkVar(Var, 1))
      ee = BkVar(Var, 1)
```

```
        tt = BkVar(Var, 2)

        cProdSched.Add Item:=howmany, key:=ee & tt
        If IsNumeric(tt) = True Then
           Worksheets("ProdSchedule").Cells(tt + 1, ii + 1) = _
           FormatNumber(howmany, 2)
        End If
        CumProd(ii) = CumProd(ii) + howmany
        row = row + 1
Loop
For E = 1 To NumAssemblies
   Worksheets("ProdSchedule").Cells(1, E + 1) = Assembly(E)
   Worksheets("ProdSchedule").Cells(hz + 2, 1) = "Total Production"
   Worksheets("ProdSchedule").Cells(hz + 2, E + 1) = CDbl(CumProd(E))
Next
Worksheets("ProdSchedule").Cells(hz + 2 + 1, 1) = "Inventory"

Debug.Print "I_"; Now; row
For E = 1 To NumAssemblies
   Worksheets("ProdSchedule").Cells(hz + 2 + 1, E + 1) = Assembly(E)
Next
row = fstrowi
Do Until row > lstrowi
        If t < 10 Then t = 0 & t
        Var = Worksheets("LPResults").Cells(row, 1)
        howmany = Worksheets("LPResults").Cells(row, 2)
        cInvSched.Add Item:=howmany, _
        key:=CStr(BkVar(Var, 1) & BkVar(Var, 2))
        row = row + 1
Loop
For E = 1 To NumAssemblies
   For t = 0 To hz
      If t < 10 Then t = 0 & t
      Worksheets("ProdSchedule").Cells(hz + 2 + t + 2, E + 1) = _
      FormatNumber(cInvSched(Assembly(E) & t), 2)
      row = row + 1
   Next
Next

row = fstrowot
If row > 0 Then
Worksheets("ProdSchedule"). _
Cells(3 * (hz + 1) + 3, 1) = "OverTime Used"
For t = 1 To hz
   If t < 10 Then t = 0 & t
   Worksheets("ProdSchedule"). _
   Cells(3 * (hz + 1) + 3 + t, 1) = t
Next
```

```
row = fstrowot
Debug.Print "OT_"; Now; row
For r = 1 To NumResources
   Worksheets("ProdSchedule").Cells(3 * (hz + 1) + 3, r + 1) = _
   resource(r)
   For t = 1 To hz
      If t < 10 Then t = 0 & t
      Var = Worksheets("LPResults").Cells(row, 1)
      howmany = Worksheets("LPResults").Cells(row, 2)
      rr = cResIdx(BkVar(Var, 1))
      tt = BkVar(Var, 2)
      If tt = 0 Then MsgBox "tt= " & tt
      Worksheets("ProdSchedule"). _
      Cells(3 * (hz + 1) + 3 + tt, rr + 1) = FormatNumber(CLng(howmany), 2)
      row = row + 1
   Next
Next
End If

Worksheets("ProdSchedule"). _
Cells(4 * (hz + 1) + 3, 1) = "Backlog"
For t = 0 To hz
   If t < 10 Then t = 0 & t
   Worksheets("ProdSchedule"). _
   Cells(4 * (hz + 1) + 3 + t + 1, 1) = t
Next

row = fstrowbl
Debug.Print "BL_"; Now; row
For E = 1 To NumAssemblies
   Worksheets("ProdSchedule").Cells(4 * (hz + 1) + 3, E + 1) = Assembly(E)
Next
Do Until row = lstrowbl + 1
   Var = Worksheets("LPResults").Cells(row, 1)
   howmany = Worksheets("LPResults").Cells(row, 2)
   If howmany >= 0 Then
      p = cAssIdx(BkVar(Var, 1))
      tt = BkVar(Var, 2)
      Worksheets("ProdSchedule"). _
      Cells(4 * (hz + 1) + 3 + tt + 1, p + 1) = _
      FormatNumber(-CLng(howmany), 2)
   End If
   row = row + 1
Loop

Worksheets("ProdSchedule"). _
Cells(2 * (hz + 2) + 1, 1) = "ExcessCap"
```

```
For t = 1 To hz
   If t < 10 Then t = 0 & t
   Worksheets("ProdSchedule"). _
   Cells(2 * (hz + 2) + 1 + t, 1) = t
Next

row = fstrowe
Debug.Print "E_"; Now; row
For r = 1 To NumResources
   Worksheets("ProdSchedule").Cells(2 * (hz + 2) + 1, r + 1) = _
   resource(r)
   For t = 1 To hz
      If t < 10 Then t = 0 & t
      Var = Worksheets("LPResults").Cells(row, 1)
      howmany = Worksheets("LPResults").Cells(row, 2)
      rr = cResIdx(BkVar(Var, 1))
      tt = BkVar(Var, 2)
      Worksheets("ProdSchedule"). _
      Cells(2 * (hz + 2) + tt + 1, rr + 1) = FormatNumber(CLng(howmany), 2)
      row = row + 1
   Next
Next

Workbooks(ActiveWorkbook.Name). _
Worksheets("ProdSchedule").Activate
Sheets("ProdSchedule").Select
Cells.Select
Range("A7").Activate
Selection.Columns.AutoFit
Range("G7").Select

'--------------------------------------------------------
'===========================================
'===========Resource Utilization==========
'===========================================
rustart = Now
'SetupTime = Worksheets("Parameters").Cells(12, 1)
Debug.Print "Resource Utilization"; Now; row
Workbooks(ActiveWorkbook.Name).Worksheets("SetupLog").Activate
Sheets("SetupLog").Select
Cells.Select
If FirstRun = "y" Then
   Selection.Delete
End If
logrow = 1
Do
   If Cells(logrow, 1) = "" Then
      Exit Do
```

```
   End If
   logrow = logrow + 1
Loop
fstlogrow = logrow
Cells(logrow, 1) = "Resource"
Cells(logrow, 2) = "Previous Setups"
Cells(logrow, 3) = "New Setups"
Cells(logrow, 5) = Now
Worksheets("SetupLog").Cells(logrow, 4) = GetOptimum("MIN")
'-------------------------------------------------------

If HeuristicOrAlgorithm = "h" Then
rp = 2
Do Until Worksheets("Parameters").Cells(rp, 3) = ""
   rp = rp + 1
Loop
Worksheets("Parameters").Cells(rp, 2) = Now
Worksheets("Parameters").Cells(rp, 3) = GetOptimum("MIN")

ElseIf HeuristicOrAlgorithm = "a" Then
rp = 2
Do Until Worksheets("Parameters").Cells(rp, 5) = ""
   rp = rp + 1
Loop
Worksheets("Parameters").Cells(rp, 4) = Now
Worksheets("Parameters").Cells(rp, 5) = GetOptimum("MIN")
End If

Workbooks(ActiveWorkbook.Name).Worksheets("ResourceUtilization").Activate
Sheets("ResourceUtilization").Select
Cells.Select
If FirstRun = "y" Then
   Selection.Delete
End If
Range("A1").Select

row = 0
rowc = 1
For r = 1 To NumResources
   Debug.Print resource(r), Now
   row = row + 1
   prow = row
   Worksheets("ResourceUtilization").Cells(row, 1) = resource(r)
   For t = 1 To hz
      setuptimes = 0
      LongestSetup = 0
      If t < 10 Then t = 0 & t
       t1 = t - 1
```

```
If t1 < 10 Then t1 = 0 & t1
row = row + 1
rowc = rowc + 1
Worksheets("ResourceUtilization").Cells(row, 1) = t
c = 1
Users = 0
used = 0
Setups = 0
For Each ensamblaje In cUsersOfResource(r)
a = cAssIdx(ensamblaje)
'For s = 1 To NumAssemblies
'For a = 1 To NumAssemblies
  c = c + 1
  Worksheets("ResourceUtilization").Cells(prow, c) = Assembly(a)
  If cRequired(resource(r) & Assembly(a)) > 0 _
  And t <= hz - lt(a) Then
    cp = cProdSched(Assembly(a) & t)
    If cp > 0 Then
      Users = Users + 1
      cr = cRequired(resource(r) & Assembly(a))
      used = used + cp * cr
      Setups = Setups + 1
      setuptimes = setuptimes + _
      cSetup(resource(r) & Assembly(a))
      If t > 1 Then
        cpm1 = cProdSched(Assembly(a) & t1)
        If cpm1 > 0 And _
        cSetup(resource(r) & Assembly(a)) > LongestSetup Then
          LongestSetup = cSetup(resource(r) & Assembly(a))
        End If
      End If
    End If
    wcp = FormatNumber(cp, 0)
    Worksheets("ResourceUtilization").Cells(row, c) = wcp
  End If
Next
cap = cCapacity(resource(r) & t)
wused = FormatNumber(used, 2)
'If (Users - cNumMach(resource(r))) > Setups Then
  'Setups = (Users - cNumMach(resource(r)))
'End If
Worksheets("ResourceUtilization").Cells(row, c + 1) = Users
Worksheets("ResourceUtilization").Cells(row, c + 3) = wused
Worksheets("ResourceUtilization").Cells(row, c + 2) = cap
If FirstRun = "n" Then
  If Setups <> Worksheets("ResourceUtilization").Cells(row, c + 4) Then
    logrow = logrow + 1
    oldsetups = Worksheets("ResourceUtilization").Cells(row, c + 4)
```

```
            Worksheets("SetupLog").Cells(logrow, 1) = resource(r)
            Worksheets("SetupLog").Cells(logrow, 2) = oldsetups
            Worksheets("SetupLog").Cells(logrow, 3) = Setups
        End If
    End If
    Worksheets("ResourceUtilization").Cells(row, c + 4) = Setups

    If used > (cap * 1.01) Then
        Worksheets("ResourceUtilization").Cells(row, c + 3).Select
        Selection.Font.ColorIndex = 3
        'With Selection.Interior
            '.ColorIndex = 6
            '.Pattern = xlSolid
        'End With
    End If
    'MsgBox ("Setups= " & Setups)
    'MsgBox ("setuptimes= " & setuptimes)
    If Setups > 0 Then
        Worksheets("ResourceUtilization").Cells(row, c + 4).Select
        Selection.Font.ColorIndex = 3
        'With Selection.Interior
            '.ColorIndex = 6
            '.Pattern = xlSolid
        'End With
        'Worksheets("Resources").Cells(rowc, 3) = _
        'Worksheets("Resources").Cells(rowc, 4) - (setuptimes) + LongestSetup
        Worksheets("Resources").Cells(rowc, "e") = _
        Worksheets("Resources").Cells(rowc, "k") - (setuptimes) + LongestSetup
    Else
        'Worksheets("Resources").Cells(rowc, 3) = _
        'Worksheets("Resources").Cells(rowc, 4)
        Worksheets("Resources").Cells(rowc, "e") = _
        Worksheets("Resources").Cells(rowc, "k")
    End If

  Next
  Worksheets("ResourceUtilization").Cells(prow, c + 1) = "NumProdUsers"
  Worksheets("ResourceUtilization").Cells(prow, c + 3) = "UsedCap"
  Worksheets("ResourceUtilization").Cells(prow, c + 2) = "AvailableCap"
  Worksheets("ResourceUtilization").Cells(prow, c + 4) = "NumSetups"
Next
'-----------------------------------------------------------
Sheets("ResourceUtilization").Select
Range("1:1").Select
Selection.Font.Bold = True
Cells.Select
Selection.Columns.AutoFit
Range("A1").Select
```

```
    GoTo nextpr
nextpr:
    Workbooks(thisWB).Worksheets("SetupLog").Activate
    Sheets("SetupLog").Select
    Cells.Select
    With Selection
        .HorizontalAlignment = xlCenter
        .VerticalAlignment = xlBottom
        .WrapText = False
        .Orientation = 0
        .AddIndent = False
        .IndentLevel = 0
        .ShrinkToFit = False
        .ReadingOrder = xlContext
        .MergeCells = False
    End With

    Range(fstlogrow & ":" & fstlogrow).Select
    Selection.Font.Bold = True
    Cells.Select
    Selection.Columns.AutoFit
    Range("A1").Select

    Workbooks(thisWB).Worksheets("ProdSchedule").Activate
    Sheets("ProdSchedule").Select
    Cells.Select
    With Selection
        .HorizontalAlignment = xlCenter
        .VerticalAlignment = xlBottom
        .WrapText = False
        .Orientation = 0
        .AddIndent = False
        .IndentLevel = 0
        .ShrinkToFit = False
        .ReadingOrder = xlContext
        .MergeCells = False
    End With

    Range("1:1").Select
    Selection.Font.Bold = True
    Cells.Select
    Selection.Columns.AutoFit
    Range("A1").Select

    'Range("1:1").Select
    Range(hz + 2 + 1 & ":" & hz + 2 + 1).Select
    Selection.Font.Bold = True
    Cells.Select
```

```
    Selection.Columns.AutoFit
    Range("A1").Select
    If HeuristicOrAlgorithm = "h" And FirstRun = "y" Then
        Worksheets("parameters").Cells(12, 1) = "n"
    End If
    If HeuristicOrAlgorithm = "h" Then MsgBox "Now Run Create LP File"
    Sheets("Parameters").Activate
    'lpformat
    Debug.Print Now, rustart, DateDiff("s", rustart, Now)
    '-------------------------------------------------------
    End
End Sub
Sub InputDataFromTextFile()
'
' ImportData Macro
' Macro recorded 12/23/2001 by Lucent
'
'
    Workbooks(ActiveWorkbook.Name).Worksheets("parameters").Activate
    Sheets("parameters").Select
    file = Cells(8, 1)
    msg = "You will get a Production Schedule from file: " & file
    If MsgBox(msg, vbOKCancel) = 2 Then
        file = InputBox("Enter path for *.int file")
    End If
    If file = "" Then End
    thisWB = ActiveWorkbook.Name
    Workbooks(ActiveWorkbook.Name).Worksheets("LPResults").Activate
    Sheets("LPResults").Select
    Cells.Select
    Selection.ClearContents
    Range("A1").Select
    'With ActiveSheet.QueryTables.Add(Connection:= _
        '"TEXT;C:\ProjectSchedule\LinearProgramming\LP-Optimizer\Win32\TASKTYPE.INT", _
        'Destination:=Range("A1"))
    'file = InputBox("Enter name of file to be imported")
    With ActiveSheet.QueryTables.Add(Connection:= _
        "TEXT;" & file, _
        Destination:=Range("A1"))
        .Name = "TASKTYPE"
        .FieldNames = True
        .RowNumbers = False
        .FillAdjacentFormulas = False
        .PreserveFormatting = True
        .RefreshOnFileOpen = False
        .RefreshStyle = xlInsertDeleteCells
        .SavePassword = False
        .SaveData = True
```

```
        .AdjustColumnWidth = True
        .RefreshPeriod = 0
        .TextFilePromptOnRefresh = False
        .TextFilePlatform = xlWindows
        .TextFileStartRow = 1
        .TextFileParseType = xlDelimited
        .TextFileTextQualifier = xlTextQualifierDoubleQuote
        .TextFileConsecutiveDelimiter = True
        .TextFileTabDelimiter = True
        .TextFileSemicolonDelimiter = True
        .TextFileCommaDelimiter = True
        .TextFileSpaceDelimiter = True
        .TextFileOtherDelimiter = ""
        .TextFileColumnDataTypes = Array(1, 1, 1)
        .Refresh BackgroundQuery:=False
    End With
    Rows("1:2").Select
    Selection.Delete Shift:=xlUp
    Cells.Select
    Selection.Sort Key1:=Range("A1"), Order1:=xlAscending, Header:=xlGuess, _
        OrderCustom:=1, MatchCase:=False, Orientation:=xlTopToBottom
End Sub
Function fstrow(st)
    Workbooks(ActiveWorkbook.Name).Worksheets("LPResults").Activate
    row = 1
    Do
        If Mid$(Cells(row, 1), 1, 2) = st Then
            fstrow = row
            Exit Do
        End If
        row = row + 1
    Loop
End Function
Function lstrow(st)
    Workbooks(ActiveWorkbook.Name).Worksheets("LPResults").Activate
    row = fstrow(st)
    Do
        If Mid$(Cells(row, 1), 1, 2) <> st Then
            lstrow = row - 1
            Exit Do
        End If
        row = row + 1
    Loop
End Function
Function BkVar(Var, dnus)
    nus = 0
    ch = 0
    Do Until nus = dnus
```

```
        ch = ch + 1
        If Mid$(Var, ch, 1) = "_" Then nus = nus + 1
    Loop
    ch = ch + 1
    Do Until Mid$(Var, ch, 1) = "_" Or ch > Len(Var)
        BkVar = BkVar & Mid$(Var, ch, 1)
        ch = ch + 1
    Loop
End Function
Function GetOptimum(goal)
    Workbooks(thisWB).Worksheets("LPResults").Activate
    row = 1
    Do
        If Cells(row, 1) = goal Then
            GetOptimum = Cells(row, 2)
            Exit Do
        End If
        row = row + 1
    Loop
End Function
Sub InitializeCapacity()
    HeuristicOrAlgorithm = Worksheets("Parameters").Cells(10, 1)
    Worksheets("parameters").Cells(12, 1) = "y"
    r = 2
    Do Until Worksheets("Resources").Cells(r, "k") = ""
        Worksheets("Resources").Cells(r, "e") = _
        Worksheets("Resources").Cells(r, "k")
        r = r + 1
    Loop
    r = 2
    If HeuristicOrAlgorithm = "h" Then
    Do Until Worksheets("Parameters").Cells(r, 3) = ""
        Worksheets("Parameters").Cells(r, 2) = ""
        Worksheets("Parameters").Cells(r, 3) = ""
        r = r + 1
    Loop
    ElseIf HeuristicOrAlgorithm = "a" Then
    Do Until Worksheets("Parameters").Cells(r, 5) = ""
        Worksheets("Parameters").Cells(r, 4) = ""
        Worksheets("Parameters").Cells(r, 5) = ""
        r = r + 1
    Loop
    End If
End Sub
Sub WritePeriodsForDemand()
    Workbooks(ActiveWorkbook.Name).Worksheets("ProductData").Activate
    hor = InputBox("Enter Number of Periods in Planning Horizon")
    For t = 1 To hor
```

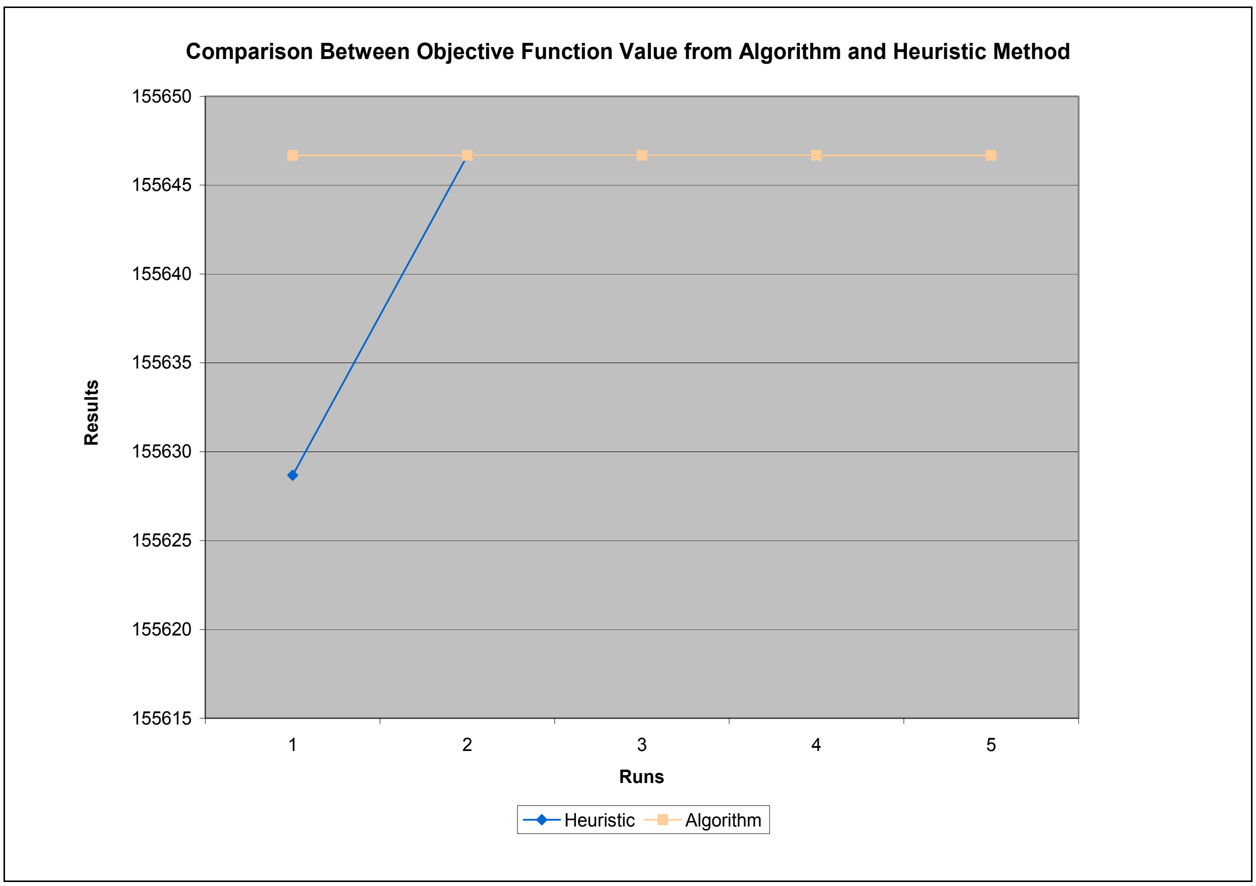
```
        t1 = t
        If t < 10 Then t1 = 0 & t
        Cells(1, 10 + t) = t1
    Next
End Sub
Sub ErasePeriodsForDemand()
    Workbooks(ActiveWorkbook.Name).Worksheets("ProductData").Activate
    hor = InputBox("Enter Number of Periods to Erase")
    For t = 1 To hor
        Cells(1, 10 + t) = ""
    Next
End Sub
```

Appendix B. Behavior of the heuristic approach

Experiment #1

State: One Product, Fighting Periods, Two Machines and 80% or lest on Capacity Utilization

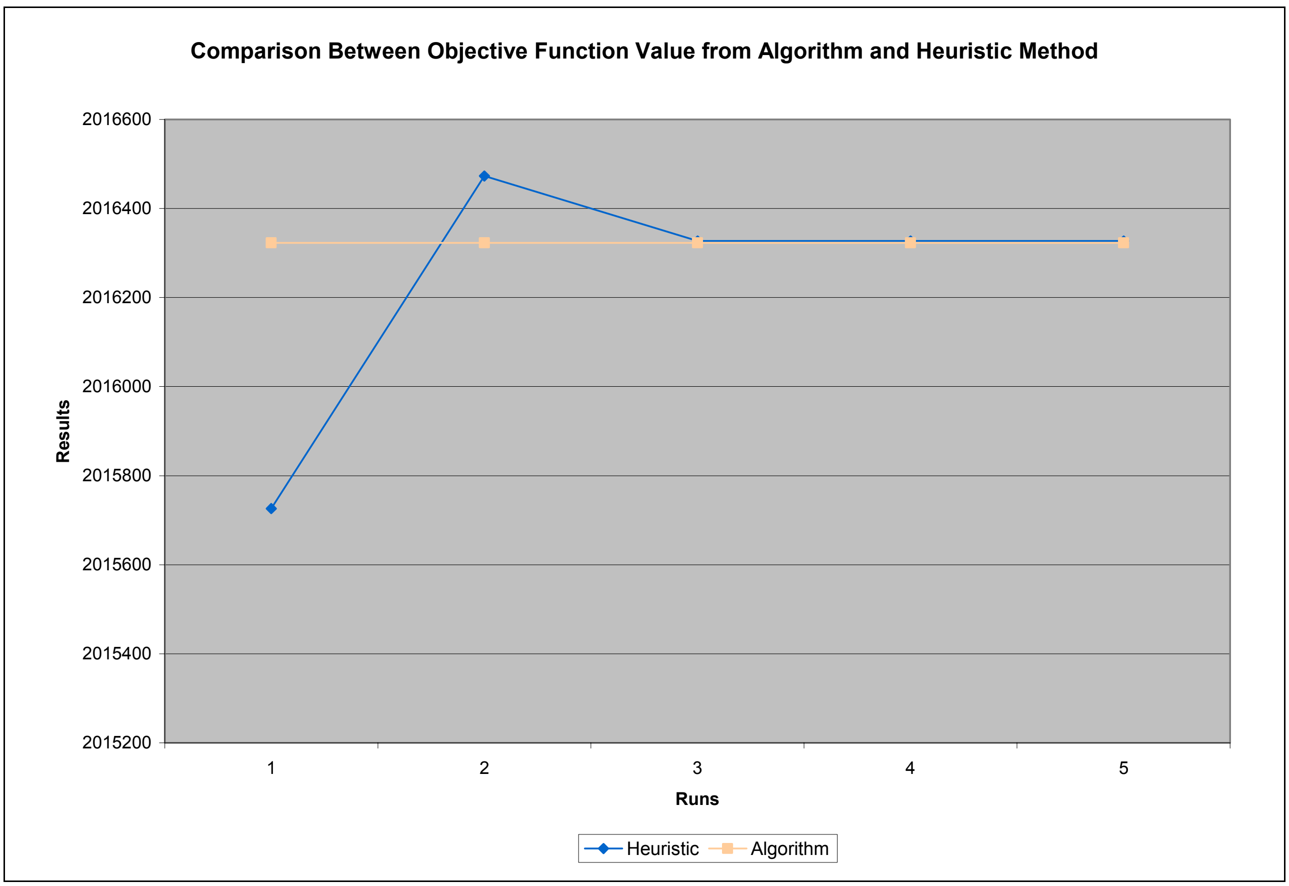
Time	Heuristic OF	Time	Algorithm OF	Algorithm ELAPSED TIME	Heuristic ELAPSED TIME
4/21/2007 13:42	155628.672	4/21/2007 13:37	155646.672	0'35.9" s	0' 0.1" s
4/21/2007 13:42	155646.672		155646.672		0' 0.1" s
4/21/2007 13:42	155646.672		155646.672		0' 0.1" s
4/21/2007 13:43	155646.672		155646.672		0' 0.1" s
4/21/2007 13:43	155646.672		155646.672		0' 0.1" s



Experiment #2

State: Three Product, Fighting Periods, Two Machines and 80% or lest on Capacity Utilization

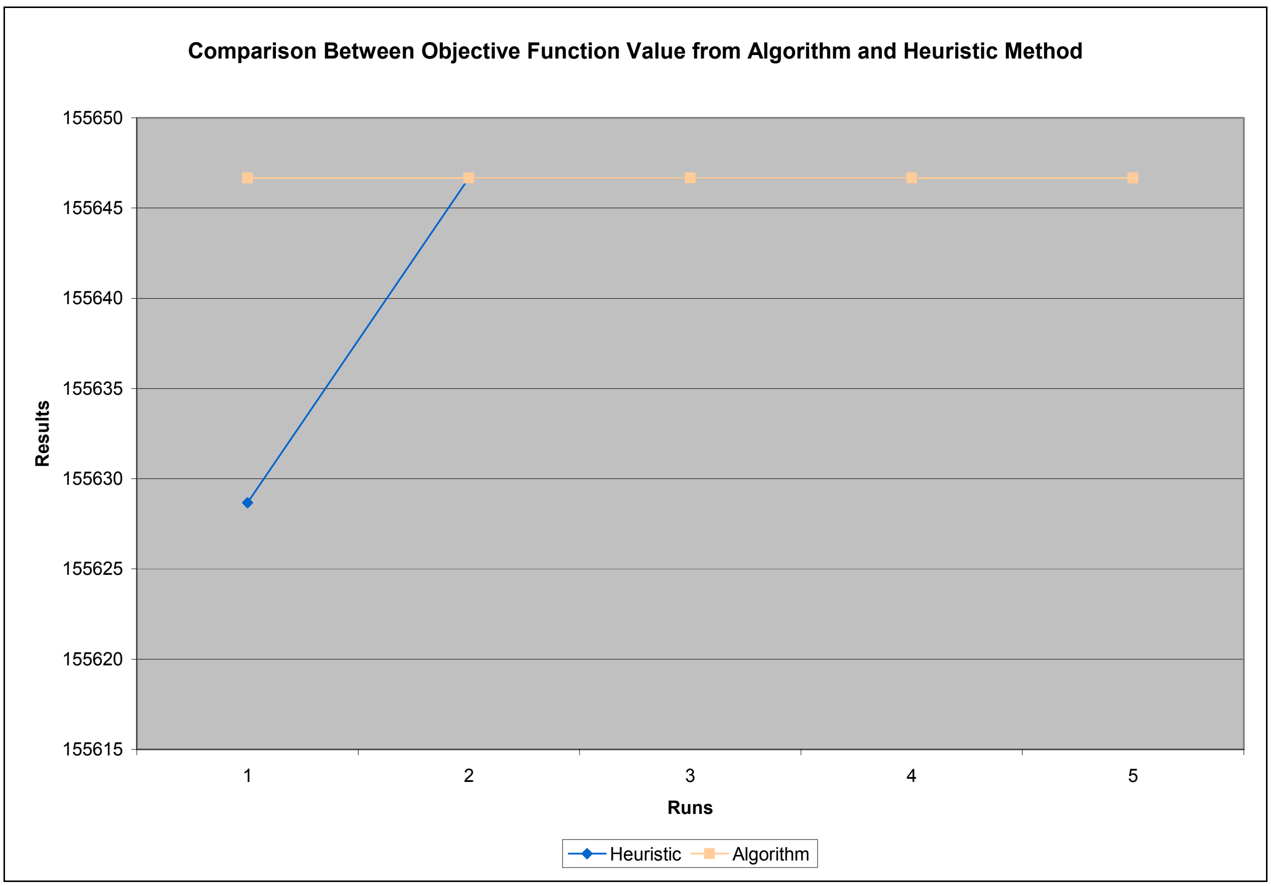
Time	Heuristic OF	Time	Algorithm OF	Algorithm ELAPSED TIME	Heuristic ELAPSED TIME
4/21/2007 14:01	2015726.500	4/21/2007 14:00	2016322.88	45'57.4" s	0' 0.1" s
4/21/2007 14:01	2016473.250		2016322.88		0' 0.1" s
4/21/2007 14:02	2016326.875		2016322.88		0' 0.1" s
4/21/2007 14:02	2016326.875		2016322.88		0' 0.1" s
4/21/2007 14:03	2016326.875		2016322.88		0' 0.1" s



Experiment #3

State: One Product, Twenty Periods, Two Machines and 80% or lest on Capacity Utilization

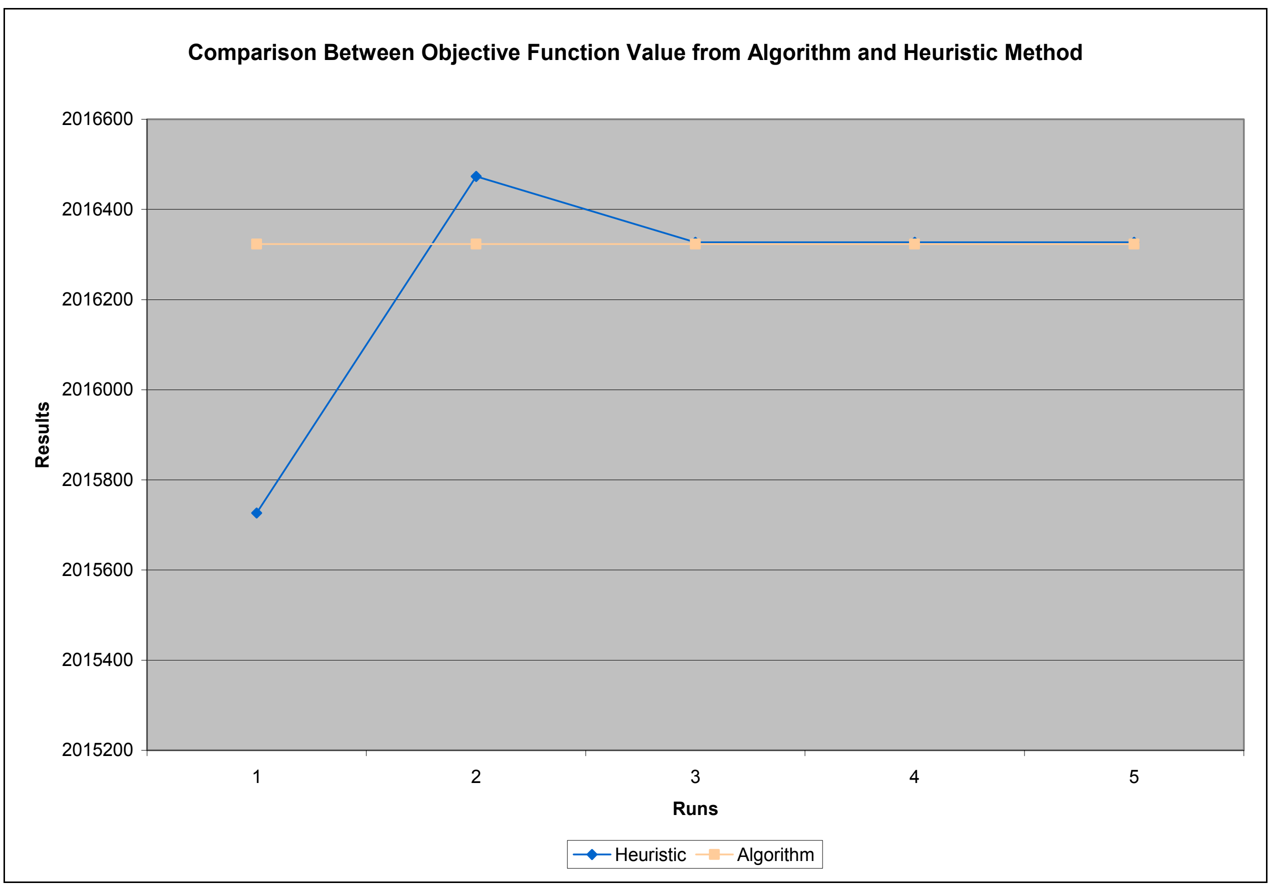
Time	Heuristic OF	Time	Algorithm OF	Algorithm ELAPSED TIME	Heuristic ELAPSED TIME
4/21/2007 13:05	155628.672	4/21/2007 13:02	155646.67	0'25.2" s	0' 0.1" s
4/21/2007 13:05	155646.672		155646.67		0' 0.1" s
4/21/2007 13:06	155646.672		155646.67		0' 0.1" s
4/21/2007 13:06	155646.672		155646.67		0' 0.1" s
4/21/2007 13:06	155646.672		155646.67		0' 0.1" s



Experiment #4

State: Three Product, Twenty Periods, Two Machines and 80% or lest on Capacity Utilization

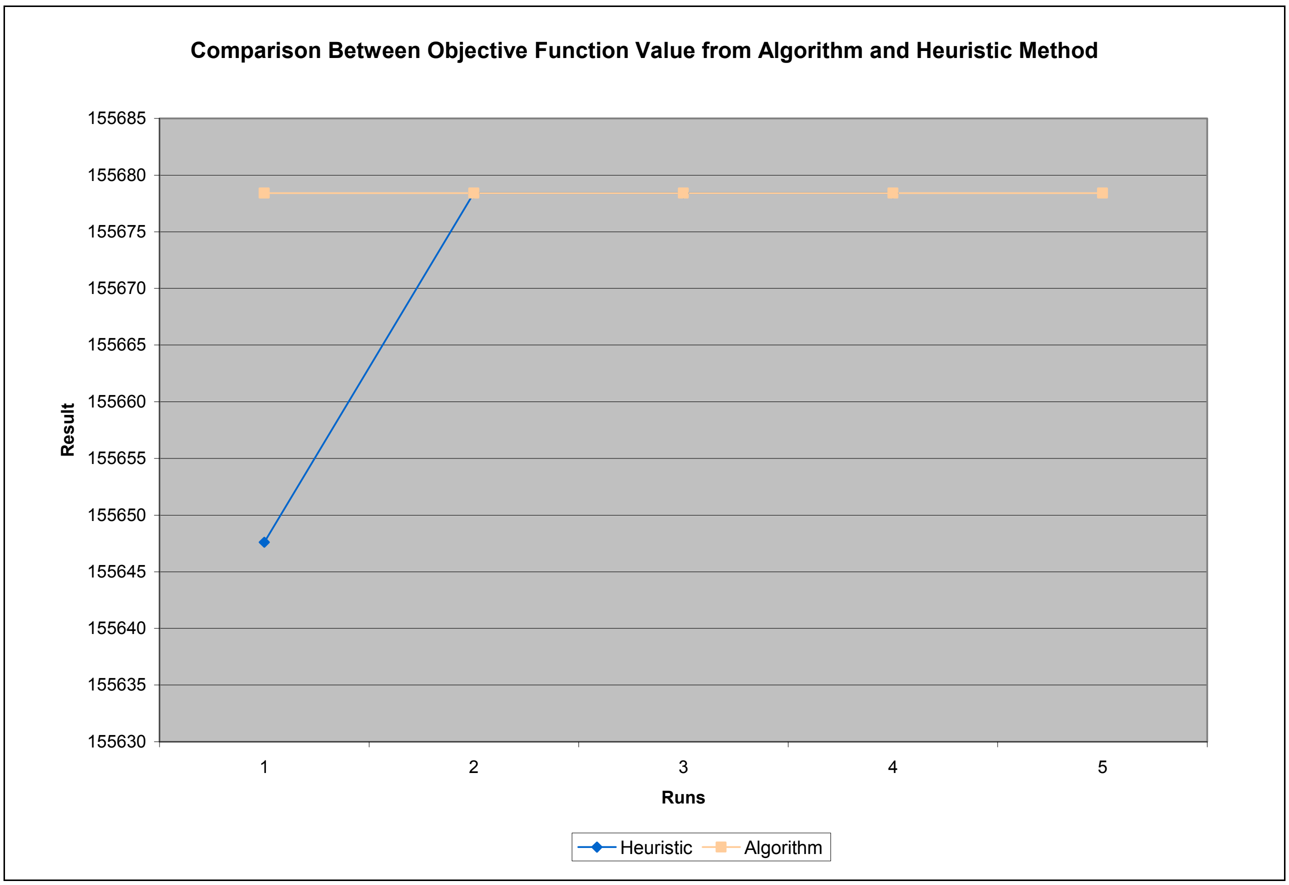
Time	Heuristic OF	Time	Algorithm OF	Algorithm ELAPSED TIME	Heuristic ELAPSED TIME
4/22/2007 11:39	2015726.500	4/22/2007 11:37	2016322.88	2h11'21.4" s	0' 0.2" s
4/22/2007 11:39	2016473.250		2016322.88		0' 0.1" s
4/22/2007 11:40	2016326.875		2016322.88		0' 0.2" s
4/22/2007 11:40	2016326.875		2016322.88		0' 0.1" s
4/22/2007 11:41	2016326.875		2016322.88		0' 0.2" s



Experiment #5

State: One Product, Fighting Periods, Three Machines and 80% or lest on Capacity Utilization

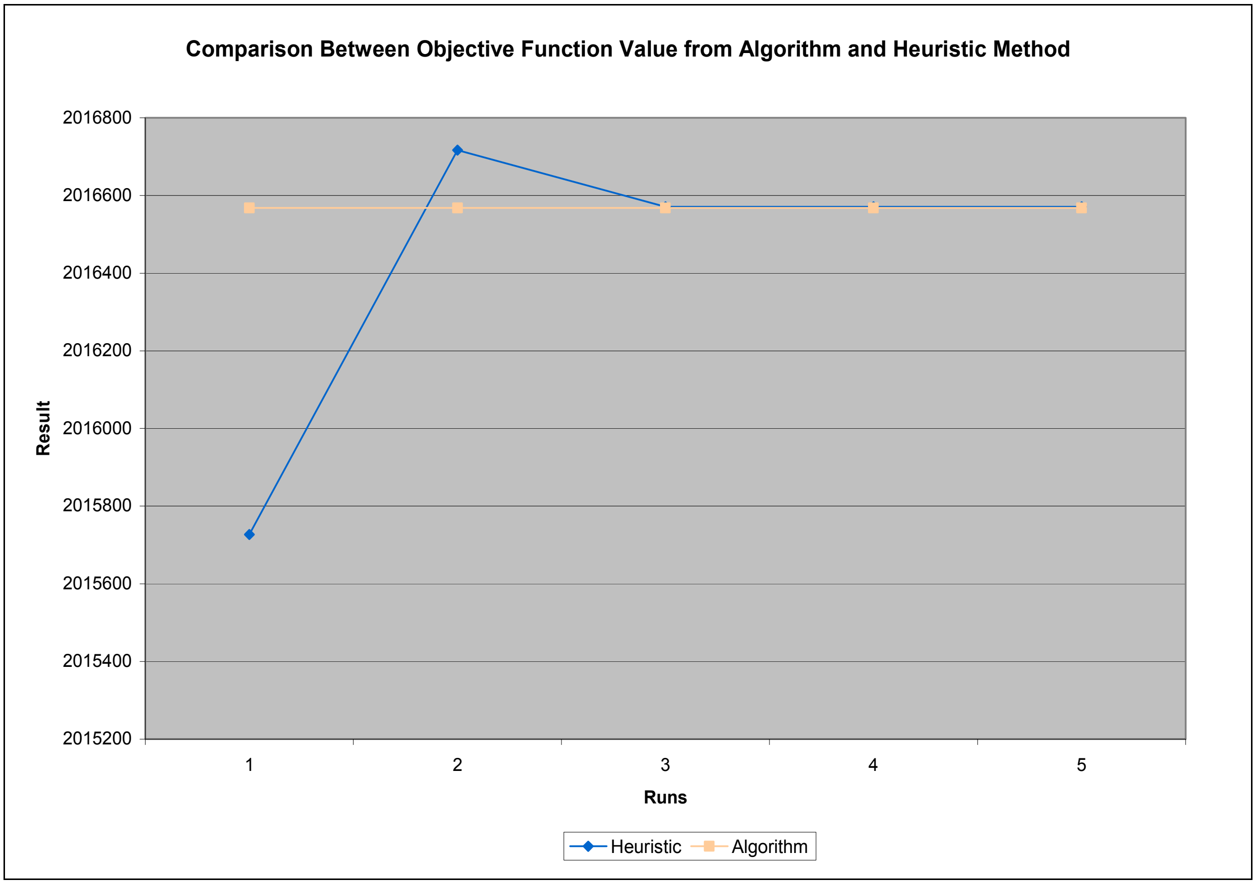
Time	Heuristic OF	Time	Algorithm OF	Algorithm ELAPSED TIME	Heuristic ELAPSED TIME
4/21/2007 12:55	155647.594	4/21/2007 12:53	155678.41	0'13.7" s	0' 0.1" s
4/21/2007 12:55	155678.406		155678.41		0' 0.1" s
4/21/2007 12:56	155678.406		155678.41		0' 0.1" s
4/21/2007 12:56	155678.406		155678.41		0' 0.1" s
4/21/2007 12:57	155678.406		155678.41		0' 0.1" s



Experiment #6

State: One Product, Twenty Periods, Two Machines and 80% or lest on Capacity Utilization

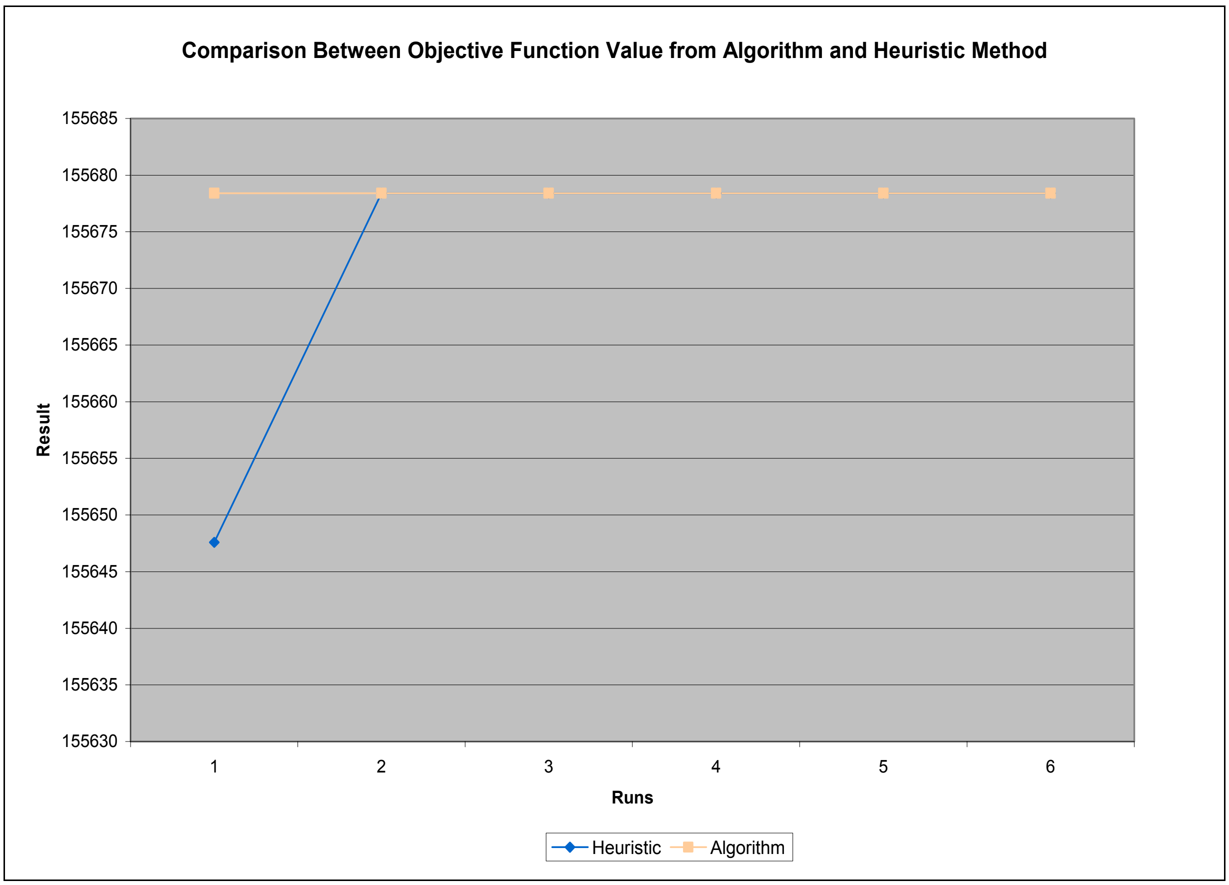
Time	Heuristic OF	Time	Algorithm OF	Algorithm ELAPSED TIME	Heuristic ELAPSED TIME
4/23/2007 14:47	2015726.750	4/23/2007 14:45	2016568.000	2h26' 7.2" s	' 0.1" s
4/23/2007 14:47	2016717.000		2016568.000		' 0.2" s
4/23/2007 14:48	2016570.625		2016568.000		' 0.2" s
4/23/2007 14:49	2016570.625		2016568.000		' 0.2" s
4/23/2007 14:49	2016570.625		2016568.000		' 0.2" s



Experiment #7

State: One Product, Twenty Periods, Two Machines and 80% or lest on Capacity Utilization

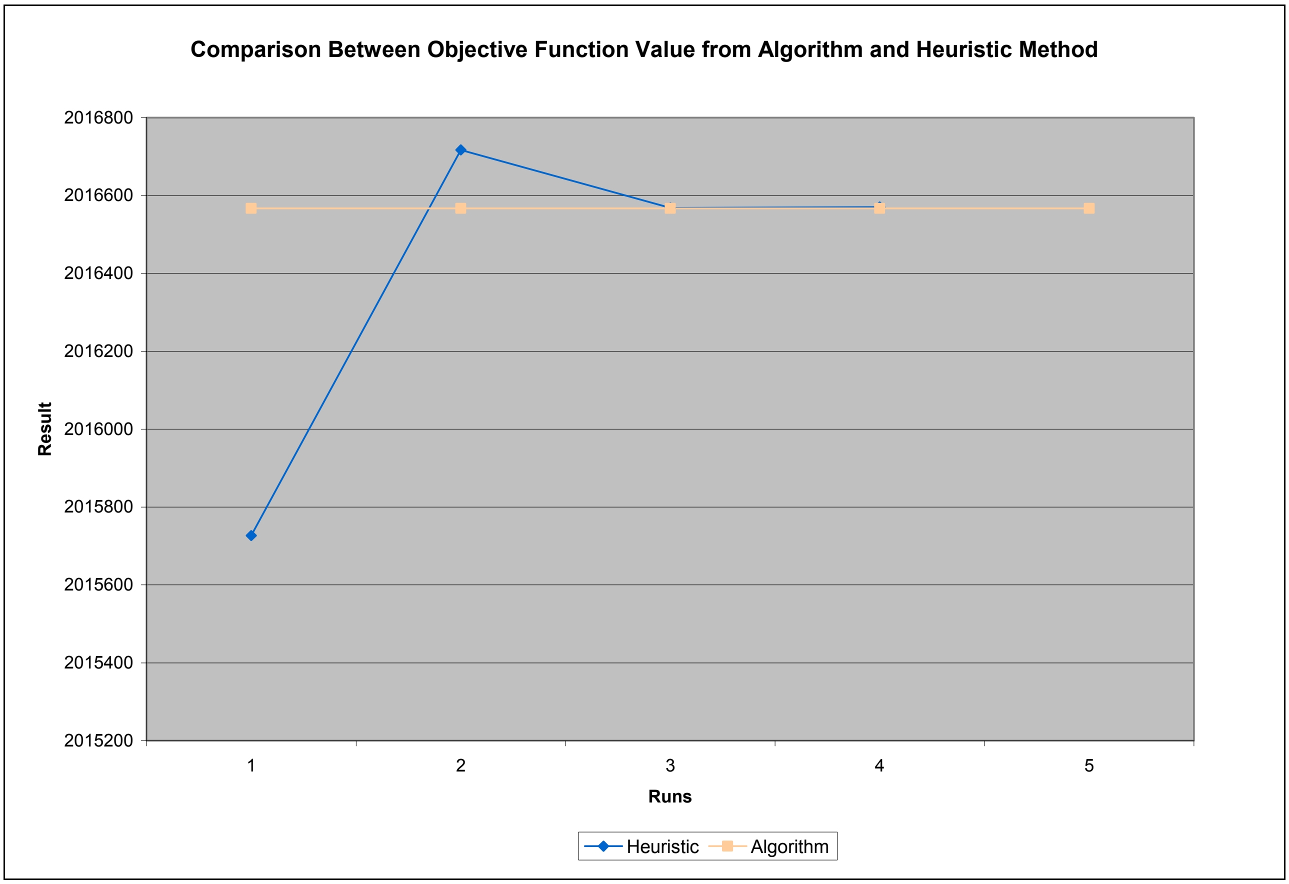
Time	Heuristic OF	Time	Algorithm OF	Algorithm ELAPSED TIME	Heuristic ELAPSED TIME
4/21/2007 12:21	155647.594	4/21/2007 12:16	155678.406	0'31.1" s	0' 0.1" s
4/21/2007 12:21	155678.406		155678.406		0' 0.1" s
4/21/2007 12:22	155678.406		155678.406		0' 0.1" s
4/21/2007 12:22	155678.406		155678.406		0' 0.1" s
4/21/2007 12:23	155678.406		155678.406		0' 0.1" s
4/21/2007 12:24	155678.406		155678.406		0' 0.1" s



Experiment #8

State: One Product, Twenty Periods, Two Machines and 80% or lest on Capacity Utilization

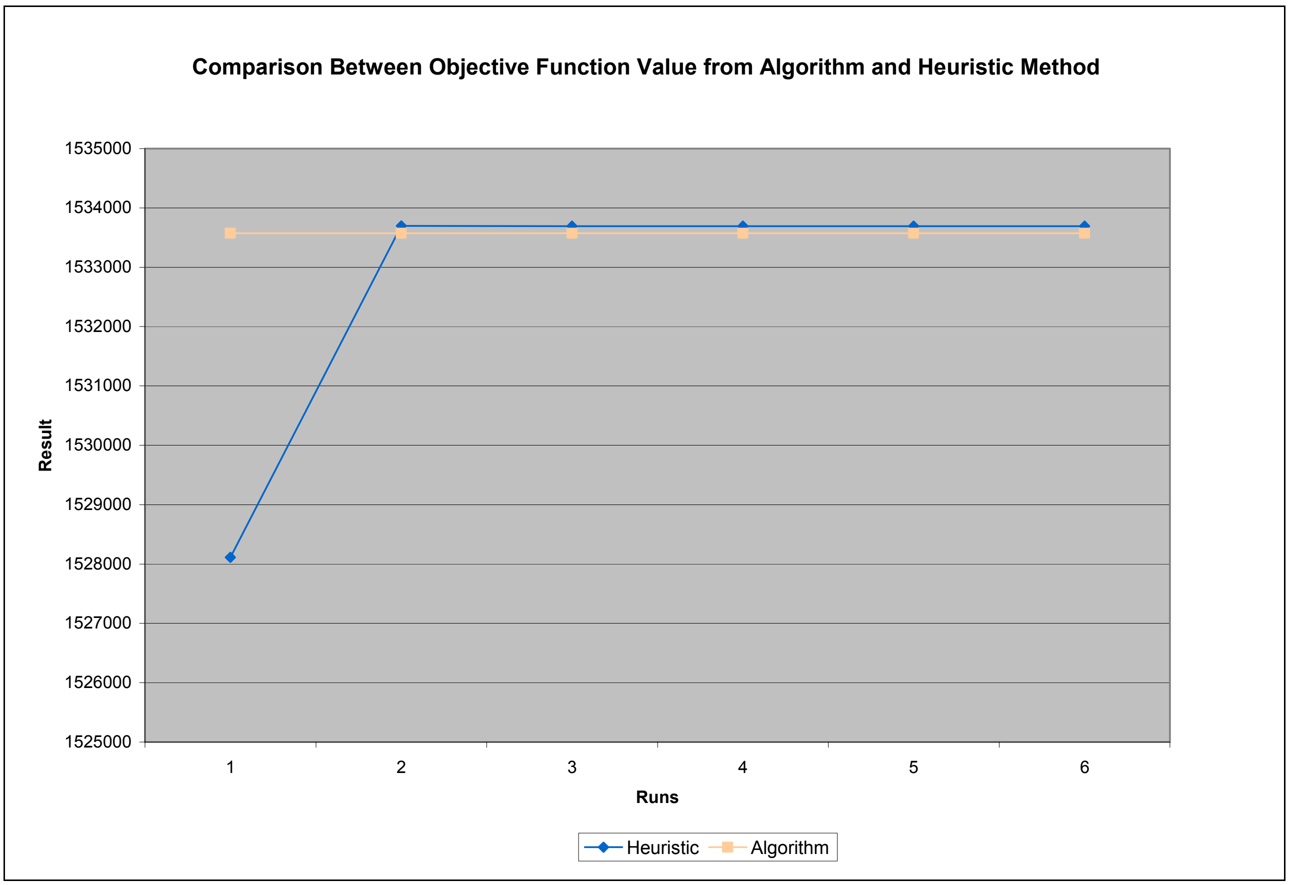
Time	Heuristic OF	Time	Algorithm OF	Algorithm ELAPSED TIME	Heuristic ELAPSED TIME
4/22/2007 11:49	2015726.750	4/22/2007 11:48	2016567.00	2h52' 7.4" s	0' 0.2" s
4/22/2007 11:49	2016717.000		2016567.00		0' 0.2" s
4/22/2007 11:50	2016568.000		2016567.00		0' 0.2" s
4/22/2007 11:50	2016570.625		2016567.00		0' 0.2" s
4/22/2007 11:51	2016568.000		2016567.00		0' 0.2" s



Experiment #9

State: One Product, Twenty Periods, Two Machines and 90% or more on Capacity Utilization

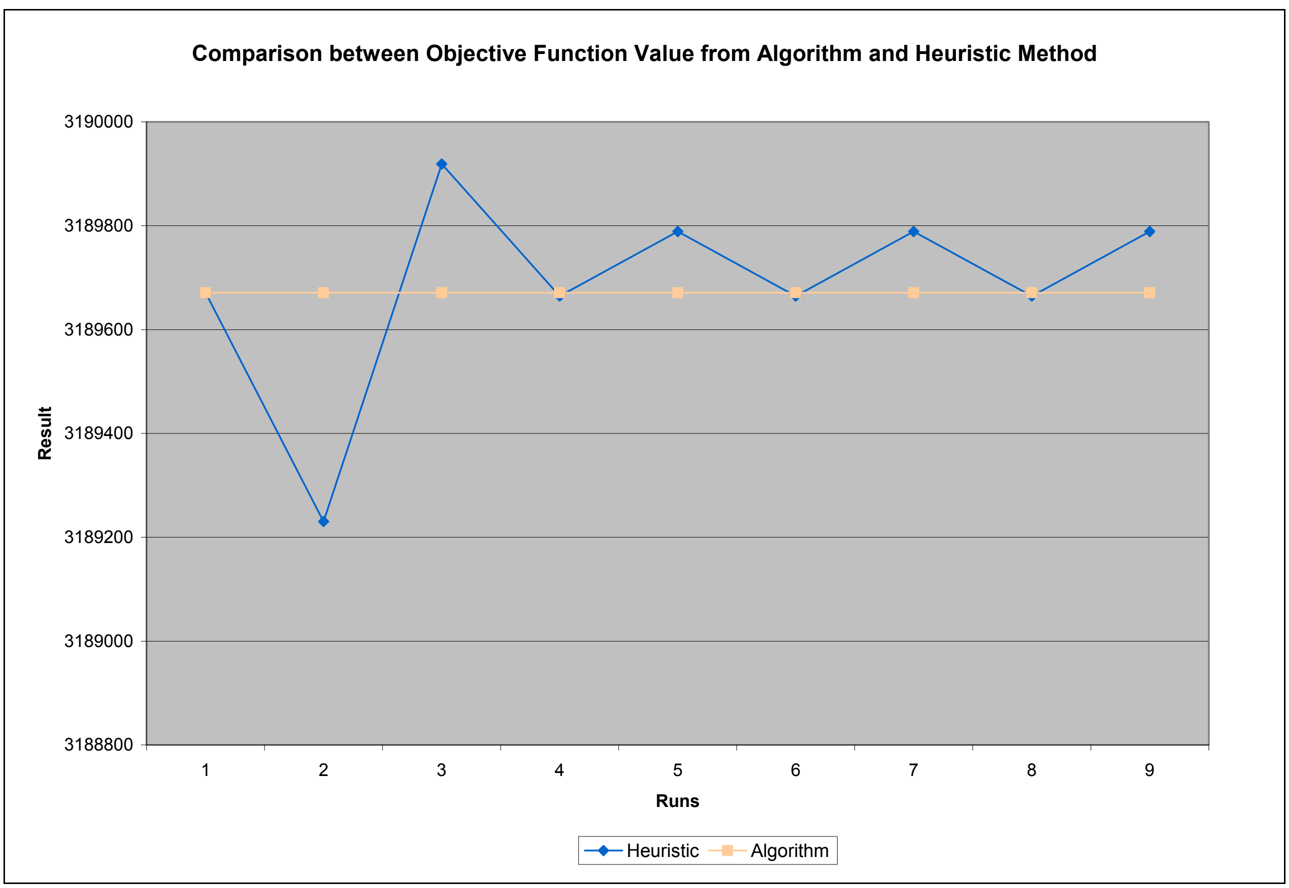
Time	Heuristic OF	Time	Algorithm OF	Algorithm ELAPSED TIME	Heuristic ELAPSED TIME
4/21/2007 12:32	1528110.750	4/21/2007 12:30	1533573.63	1h 8'26.0" s	0' 0.1" s
4/21/2007 12:32	1533697.875		1533573.63		0' 0.1" s
4/21/2007 12:33	1533692.500		1533573.63		0' 0.1" s
4/21/2007 12:33	1533692.500		1533573.63		0' 0.1" s
4/21/2007 12:33	1533692.500		1533573.63		0' 0.1" s
4/21/2007 12:34	1533692.500		1533573.63		0' 0.1" s



Experiment #10

State: One Product, Twenty Periods, Two Machines and 90% or more on Capacity Utilization

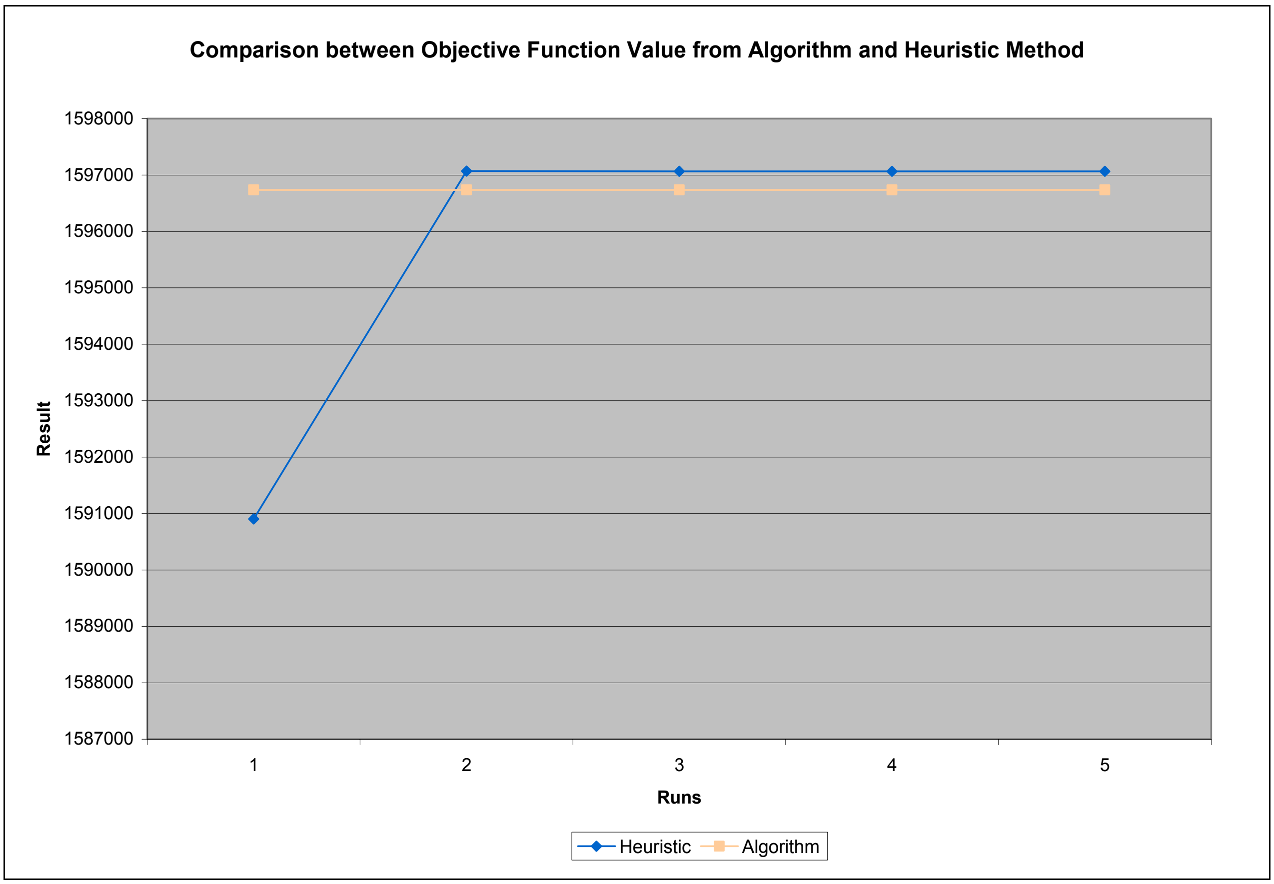
Time	Heuristic OF	Time	Algorithm OF	Algorithm ELAPSED TIME	Heuristic ELAPSED TIME
4/21/2007 12:06	3189671.000	4/21/2007 12:05	3189671.00	8h31' 0.8" s	0' 0.1" s
4/21/2007 12:07	3189230.250		3189671.00		0' 0.1" s
4/21/2007 12:08	3189919.000		3189671.00		0' 0.1" s
4/21/2007 12:08	3189664.750		3189671.00		0' 0.1" s
4/21/2007 12:09	3189789.000		3189671.00		0' 0.1" s
4/21/2007 12:09	3189664.750		3189671.00		0' 0.1" s
4/21/2007 12:10	3189789.000		3189671.00		0' 0.1" s
4/21/2007 12:10	3189664.750		3189671.00		0' 0.1" s
4/21/2007 12:11	3189789.000		3189671.00		0' 0.1" s



Experiment #11

State: One Product, Twenty Periods, Two Machines and 80% or lest on Capacity Utilization

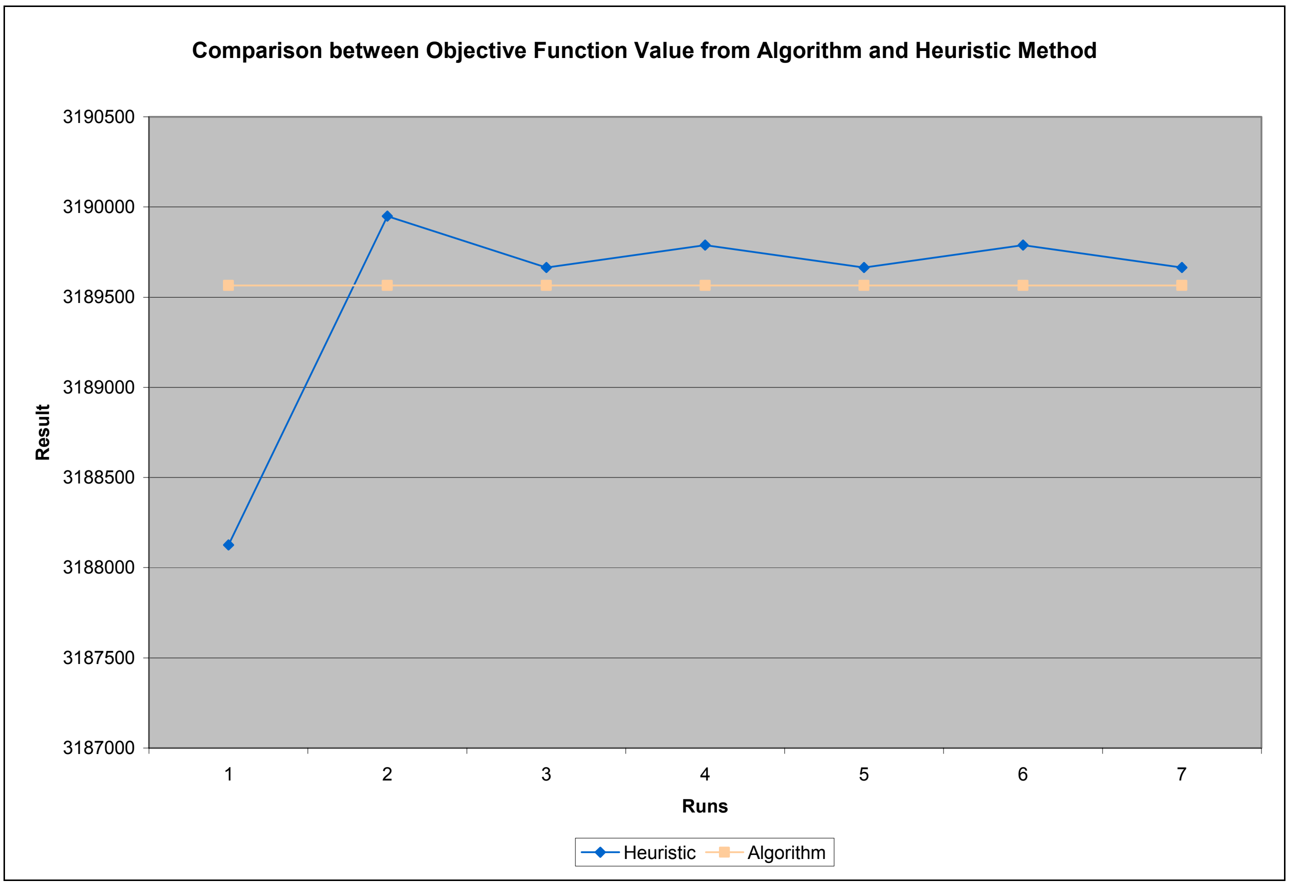
Time	Heuristic OF	Time	Algorithm OF	Algorithm ELAPSED TIME	Heuristic ELAPSED TIME
4/26/2007 8:40	1590904.500	4/26/2007 8:38	1596736.50	1d 5h 9'20.0" s	0' 0.1" s
4/26/2007 8:40	1597072.250		1596736.50		0' 0.1" s
4/26/2007 8:41	1597066.875		1596736.50		0' 0.1" s
4/26/2007 8:41	1597066.875		1596736.50		0' 0.1" s
4/26/2007 8:42	1597066.875		1596736.50		0' 0.1" s



Experiment #12

State: One Product, Twenty Periods, Two Machines and 90% or lest on Capacity Utilization

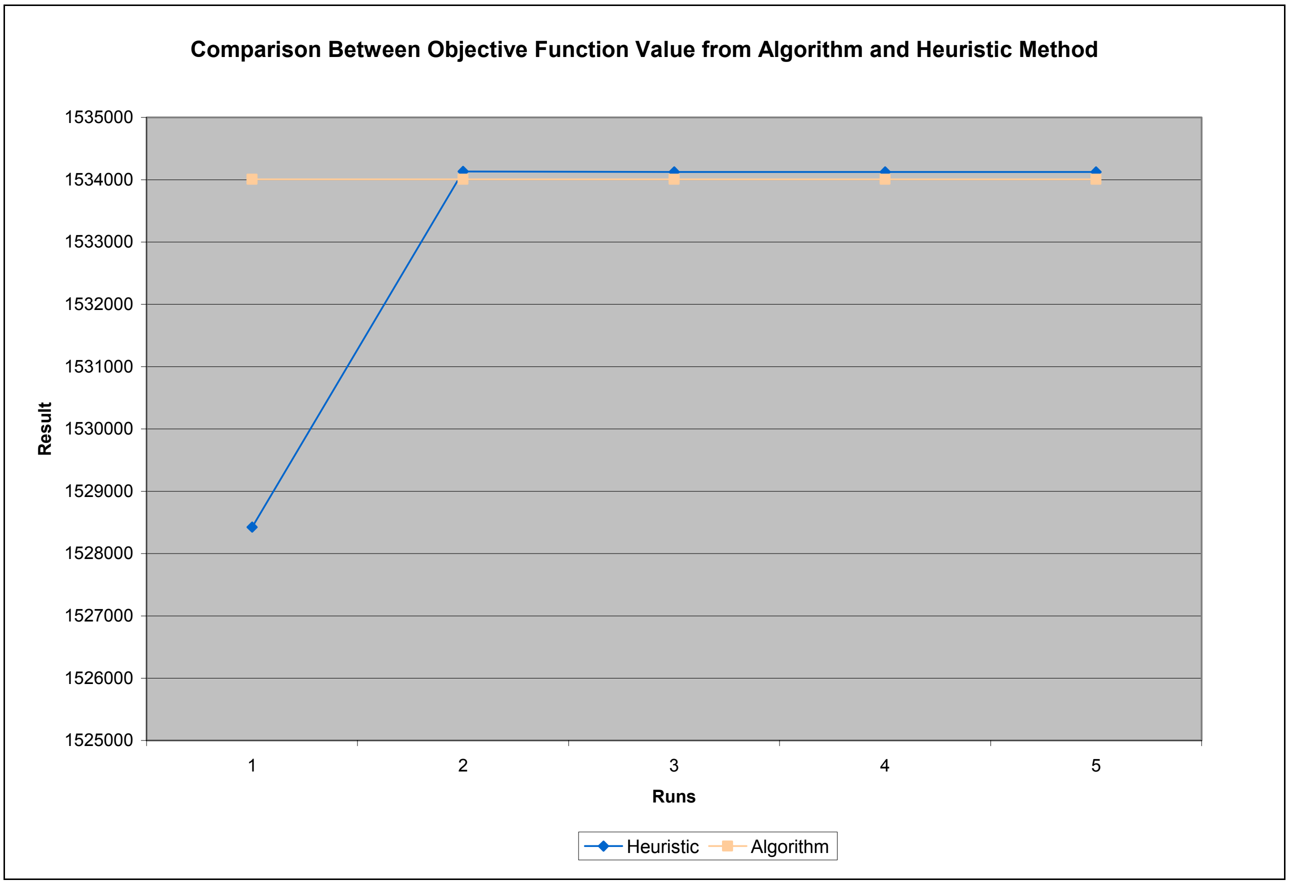
Time	Heuristic OF	Time	Algorithm OF	Algorithm ELAPSED TIME	Heuristic ELAPSED TIME
4/21/2007 11:48	3188126.500	4/21/2007 11:46	3189564.75	17h49'44.3" s	0' 0.1" s
4/21/2007 11:49	3189949.000		3189564.75		0' 0.2" s
4/21/2007 11:49	3189664.750		3189564.75		0' 0.1" s
4/21/2007 11:50	3189789.000		3189564.75		0' 0.2" s
4/21/2007 11:51	3189664.750		3189564.75		0' 0.1" s
4/21/2007 11:51	3189789.000		3189564.75		0' 0.2" s
4/21/2007 11:52	3189664.750		3189564.75		0' 0.1" s



Experiment #13

State: One Product, Twenty Periods, Two Machines and 90% or more on Capacity Utilization

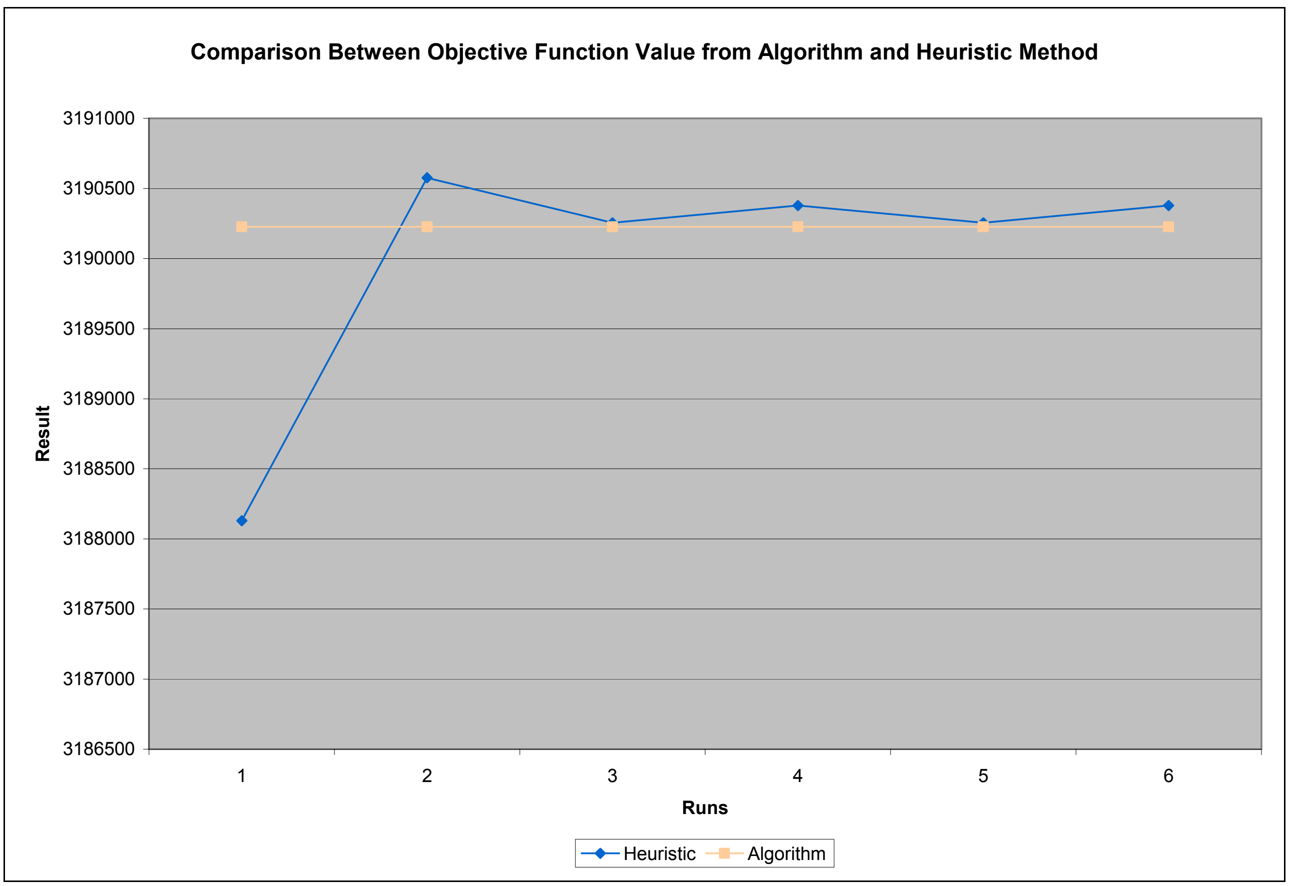
Time	Heuristic OF	Time	Algorithm OF	Algorithm ELAPSED TIME	Heuristic ELAPSED TIME
4/21/2007 12:45	1528426.000	4/21/2007 12:43	1534007.25	22h35'53.0" s	0' 0.1" s
4/21/2007 12:45	1534131.500		1534007.25		0' 0.1" s
4/21/2007 12:46	1534126.125		1534007.25		0' 0.1" s
4/21/2007 12:46	1534126.125		1534007.25		0' 0.1" s
4/21/2007 12:46	1534126.125		1534007.25		0' 0.1" s



Experiment #14

State: One Product, Twenty Periods, Two Machines and 80% or lest on Capacity Utilization

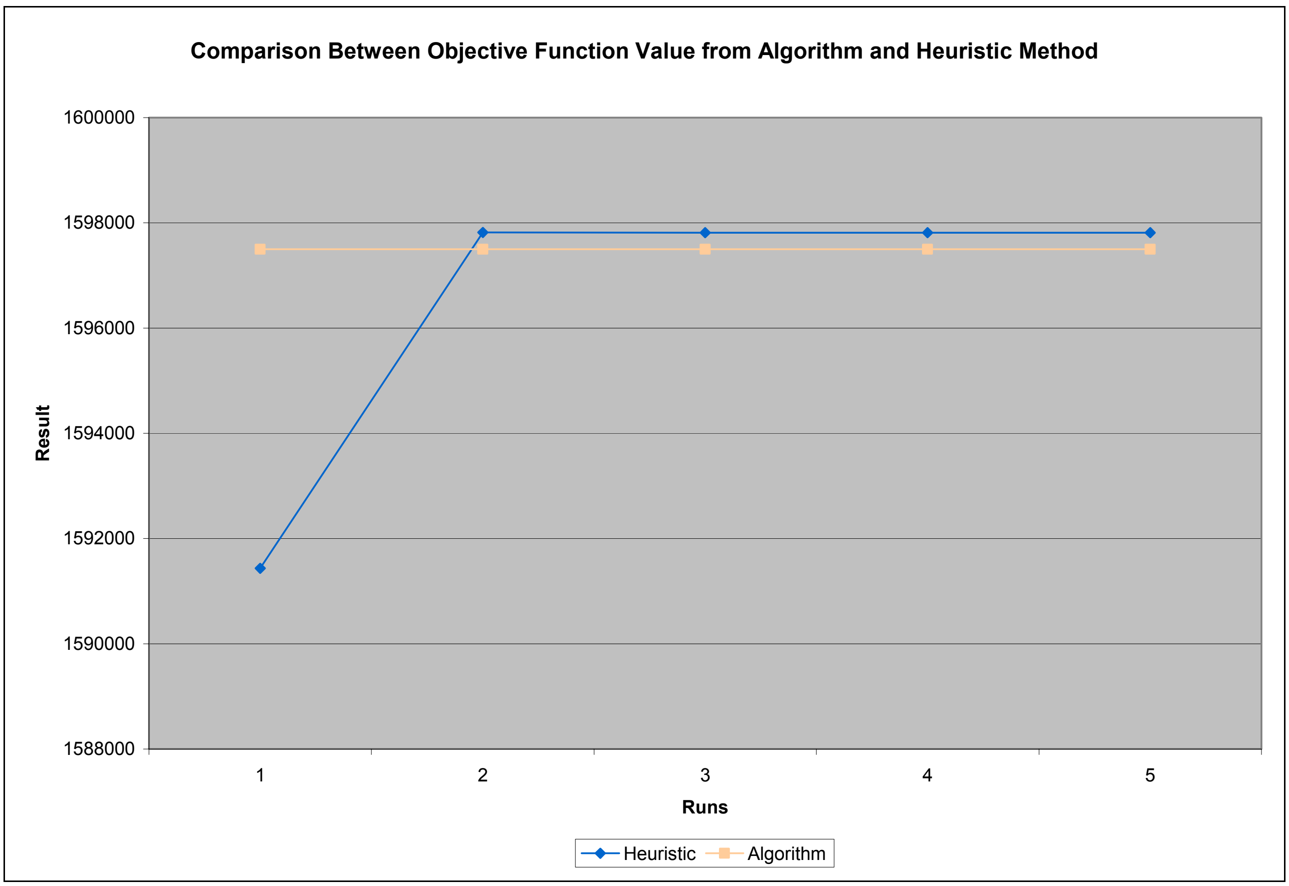
Time	Heuristic OF	Time	Algorithm OF	Algorithm ELAPSED TIME	Heuristic ELAPSED TIME
4/23/2007 15:38	3188130.000	4/23/2007 15:34	3190226.00	3d 1h56' 8.0" s	0' 0.1" s
4/23/2007 15:39	3190576.750		3190226.00		0' 0.1" s
4/23/2007 15:39	3190254.750		3190226.00		0' 0.1" s
4/23/2007 15:40	3190379.000		3190226.00		0' 0.1" s
4/23/2007 15:40	3190254.750		3190226.00		0' 0.1" s
4/23/2007 15:40	3190379.000		3190226.00		0' 0.1" s



Experiment #15

State: One Product, Twenty Periods, Two Machines and 80% or lest on Capacity Utilization

Time	Heuristic OF	Time	Algorithm OF	Algorithm ELAPSED TIME	Heuristic ELAPSED TIME
4/24/2007 7:57	1591436.375	4/24/2007 7:54	1597500.00	3d18h23' 9.5" s	0' 0.1" s
4/24/2007 7:58	1597818.500		1597500.00		0' 0.1" s
4/24/2007 7:58	1597813.125		1597500.00		0' 0.1" s
4/24/2007 7:58	1597813.125		1597500.00		0' 0.1" s
4/24/2007 7:59	1597813.125		1597500.00		0' 0.1" s



Experiment #16

State: One Product, Twenty Periods, Two Machines and 80% or lest on Capacity Utilization

Time	Heuristic OF	Time	Algorithm OF	Algorithm ELAPSED TIME	Heuristic ELAPSED TIME
4/30/2007 9:53	3188130.00	4/30/2007 9:52	3190217.00	9d20h29'17.5" s	0' 0.2" s
4/30/2007 9:54	3190576.75		3190217.00		0' 0.2" s
4/30/2007 9:54	3190254.75		3190217.00		0' 0.2" s
4/30/2007 9:55	3190379.00		3190217.00		0' 0.2" s
4/30/2007 9:55	3190254.75		3190217.00		0' 0.2" s
4/30/2007 9:56	3190379.00		3190217.00		0' 0.2" s