A SCHEDULING PROBLEM IN COPPER WIRE INDUSTRY

by

Noyan Zümbülcü

B.S. in Management Engineering, Istanbul Technical University, 2001

Submitted to the Institute for Graduate Studies in
Science and Engineering in partial fulfillment of
the requirements for the degree of
Master of Science

Graduate Program in Industrial Engineering
Boğaziçi University
2010

ACKNOWLEDGEMENTS

I would like to gratefully acknowledge the enthusiastic supervision, invaluable guidance and help of Professor Ali Tamer Ünal during this dissertation. His expertise, understanding and patience added considerable value to my graduate experience. I appreciate his vast knowledge and skill in many areas and his assistance in writing reports.

I must also acknowledge Mr. Özgür Gültekin from Erbakır Elektrolit, where the problem in this thesis is inspired from. He was always helpful during data collection and problem description phases with both his deep knowledge of the general processes of Erbakır and his great experience in planning procedures.

Finally, I am forever indebted to my parents for their understanding, endless patience and encouragement when it was most required.

ABSTRACT

A SCHEDULING PROBLEM IN COPPER WIRE INDUSTRY

This thesis addresses the problem of scheduling jobs on two stage unrelated parallel machines with job sequence dependent setup times so as to minimize the defined objective function in copper-wire industry. The study of the problem stemmed from a real copper-wire manufacturer company, Erbakır Elektrolit. The mathematical representation of the problem considering the industry specific parameters, decision variables and constraints is presented. Since the presented problem is NP-hard due to several factors, a heuristic algorithm is proposed and tested with real life data sets. The proposed heuristic algorithm is coded in company's live system using SAP's internal programming language ABAP. The heuristic is composed of two stages, which are most difficult to schedule in Erbakır Elektrolit in real life. At first stage, simulated annealing technique is implemented after identifying an initial solution which also considers the properties of the problem. Parameters of the simulated annealing approach are tuned with one set of data and several test runs. At the second phase, the second stage of the problem is solved again with another heuristic approach given the input of the solution in the first stage. Heuristics are run with two sets of comparison data and the results are compared with the planners plan in the system. Results of the proposed heuristic overcome the result of the planner.

ÖZET

BAKIR TEL ENDÜSTRİSİNDE BİR ÇİZELGELEME PROBLEMİ

Bu tez çalışması, bakır-tel endüstrisinde belirlenen amaç fonksiyonunu minimize etmek hedefiyle sıra bağımlı hazırlık sürelerini dikkate alan iki seviyeli, bağımsız parallel makine problemini içermektedir. Bu çalışmada belirtilen konu, Erbakır Elektrolit'teki gerçek hayat probleminden türetilmiştir. Sektöre özgü parametreleri, karar değişkenlerini ve kısıtları içeren problemin matematiksel gösterimi de bu çalışma içerisinde yer almaktadır. Sunulan problem lineer olmayan kısıtlar içermesi ve yüksek sayıda makine-iş kombinasyonuna sahip olması gibi nedenlerle NP-Zor olarak nitelenmektedir. Bu sebeple bir sezgisel algoritma tasarlanmış, bu algoritma gerçek hayattaki datalar ile test edilmiştir. Önerilen sezgisel algoritma, şirketin canlı sisteminde SAP'nin dahili yazılım dili olan ABAP ile kodlanmıştır. Sezgisel algoritma, Erbakır için gerçek hayatta planlanması en zor olan iki aşamayı içermektedir. Birinci bölümde, problemin ilk aşamasındaki özel kısıtlar da düşünülerek elde edilmiş olan çözümü baz alacak şekilde simulated annealing tekniği uygulanmıştır. Simulated annealing yaklaşımında kullanılacak olan parametreler bir set veri ve pek çok test çalıştırması sonucunda belirlenmiştir. İkinci bölümde ise, problemin ikinci aşaması, ilk aşamadaki sonucu girdi kabul edecek şekilde geliştirilen bir sezgisel yöntem yardımı ile çözülmüştür. Sezgisel yöntemler, iki set gerçek veri ile test edilmiş ve planlamacının sistemdeki planı ile karşılaştırılmıştır. Önerilen sezgisel yöntemin sonuçları, bu plandan daha başarılı olmuştur.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS / ABBREVIATIONS

a_{kj}^{s}	Machine assignment parameter of spool type k machine j at stage s
AR_{j}^{1}	Average radius on machine *j*
AT_{i}^{1}	Average tardiness of job *i*
b_{i}	Component count of job *i*
B_{ij}^{1}	Component count of job *i* if it is assigned to machine *j* at stage *1*
C_{jt}^{s}	Capacity of machine *j* at stage s at period *t*
CS_{i}	Spool count of the customer requirement of job i
d_{i}	Due date of job *i*
dl_{i}	Location difference between stages for job *i*
$Dmax_{i}^{1}$	Maximum due date of job *i* at stage *1*
$Dmin_{i}^{1}$	Minimum due date of job *i* at stage *1*
E_{ij}^{1}	Machine eligibility parameter
ET_{i}^{s}	End time of job *i* at stage *s*
F_{j}^{s}	First available time in machine *j* in stage *s*
i	Job index, $i = 1,2,3, \ldots, N$
j	Machine index, $j = 1,2,3, \ldots, m^{s}$
k	Spool type, $k = 1,2,3,\ldots,K$
l	Locations, $l = 1,2,3$
L_{j}^{s}	Location of machine *j* at stage s
$\mathrm{LP}_{\mathrm{i}}^{1-2}$	Overall location change penalty of job *i*
LP_{ij}^{2}	Location change penalty of job *i* when assigned to machine *j*
M	A large number
$MAXV_{ij}^{2}$	Number of maximum parallel jobs in stage *2* when job *i* is on machine *j* in stage *1*
MF_{i}^{1}	Machine flexibility index of job *i*
MP_{ij}^{1}	Maximum number of parallel split in stage *2* when job *i* is on machine *j* in stage *1*.
m^{s}	Number of unrelated parallel machines at stage *s*
MS_{ij}^{1}	Machine selection index of job *i* and machine *j*

NM_l^2	Number of machines according to locations in stage 2
PA_{ijv}^2	Possible parallel job parameter in stage *2*
p_i	Number of parallel machines for job *i*
PM_{ij}^2	Machine penalty index when the job *i* is assigned to machine *j*
r_i	Radius of job *i*
r_{jt}^{max}	Maximum radius on machine *j* at period *t*
r_{jt}^{min}	Minimum radius on machine *j* at period *t*
RT_i^2	Release time of job *i* at stage *2*
s	Stage index, $s = 1,2$
SC_i^1	Sequence dependent setup cost for job *i* at stage *1*
SC_k^{max}	Maximum weight limit of spool type *k*
SC_k^{min}	Minimum weight limit of spool type *k*
S_{ik}^{1-2}	Spool transfer variable for job *i* and spool *k* between stage 1 and 2
S_{ikjv}^{1-2}	Spool transfer parameter between stages
SL_{ij}^1	Slack of job *i* in stage *s* when it is assigned to machine *j*
$SLmax_i^1$	Slack of job *i* to maximum due date at stage *1*
$SLmin_i^1$	Slack of job *i* to minimum due date at stage *1*
SP_i^2	Overall spooler / component count compatibility penalty of job *i*
SP_{ij}^2	Spooler / component count compatibility penalty of job *i* when assigned to machine *j*
SS_{ijv}^1	Spool size of job *i* on machine *j* in stage 1 with *v* parallel machine in stage *2*
ST_i^s	Start time of job *i* at stage *s*
t	Time period, $t = 1,2,3,...,T$
T_i	Tardiness of job *i*
tw_i	Total weight of job *i*
u_{ij}^s	Processing time of job *i* on machine *j* at stage s
v	Number of parallel machines at stage *2*, $v = 1,2,3, . . .,V$
v_j^2	Spooler count of machine *j* at stage *2*
x_{ijt}^s	Machine assignment variable for job *i* on machine *j* at period *t* at stage *s*

Cmax	Machine number of the job that has maximum available machine.
LOCAVG	Average unit machine load over three physical locations
LOCCOST	Total unit machine load cost
MW1	Weight of tardiness factor in objective function
MW2	Weight of setup cost factor in objective function
MW3	Weight of location change factor in objective function
W1	Weight of setup cost - Initial Solution – Stage *1*
W2	Weight of slack - Initial Solution – Stage *1*
W3	Weight of setup costs – Corrective Solution – Stage *1*
W4	Weight of slack to minimum due date - Corrective Solution – Stage *1*
W5	Weight of slack to maximum due date - Corrective Solution – Stage *1*
W6	Weight of location change cost - Corrective Solution – Stage *1*
W7	Weight of roller/component count incompatibility - Corrective Solution – Stage *1*
W8	Weight of location change cost - Solution – Stage *2*
W9	Weight of roller/component count incompatibility - Solution – Stage *2*
W10	Weight of slack/tardiness to due date - Solution – Stage *2*

1. INTRODUCTION

In this study, we worked on a scheduling problem of a flow-shop production in copper-wire industry. Copper-wire industry has some specific constraints in planning due to its unique type of production.

The flow structure of the copper-wire production can be described as a flow-shop production. There are several consequent operations that a product should pass and this path is defined and identified for each product beforehand. However, since the variety of products is high and the expectations of the customers change towards special products rather than standard ones, which increase the number of special products, neither all the products passes all production steps nor the sequence of the operations are the same for all of them. For example, tinning process is only valid for a certain product group, or the operation sequence of a product group contains twisting operation more than once. This property of the production increases the complexity of the scheduling problem and changes the structure from a standard flow-shop scheduling problems to a more complex one.

Copper production has also its own constraints and conditions apart from the complexity of the production. The raw material of copper-wire is copper ore and both the price itself and the fluctuation on the unit market price is very high and unpredictable. Therefore, purchasing of the copper ore does not depend on the customer demand; rather the purchase of the ore is executed according to the current market price of ore and the economical forecasts. One other important factor that affects the purchasing time of the raw material is the distance of the vendors. Most of the vendors of ore are in South America and shipments between locations are done periodically. To overcome these factors, planning of the raw material purchase is done based on forecasts and forecast errors either increases the stock keeping costs or stock-out periods happen.

On the other hand, customer demand is not stable in copper wire industry. Customers are requesting ad-hoc bases and mostly it is not easy to predict the actual demand. Also, physical production volume is high and the product is heavy in weight and it is difficult to transport from one location to another. Therefore, alternative parallel machines exist in production area for every stage in the production and they are located in separate physical locations. Each location contains machines of most of the stages. In production, because of the problems in transporting the spools between stages, it is preferred to produce all the stages in one physical location, even if machines in different locations are alternative to each other for a single product.

There are other planning related issues in copper-wire, some of which are:

- Alternative non-identical parallel machines,
- Sequence dependent setup durations,
- Overlapping between operations and transfer lot sizes,
- Eligibility constraints on machines (separate products are produced in separate alternatives),
- Alternative bill of materials and constraints according to the alternative,
- Due date and tardiness constraints.

These issues are identified in the related section.

In this study, we will analyze the properties and scheduling constraints in copper-wire production. From a business point of view, the production scheduling problem can be decomposed into three main sections. According to the production planning responsible in copper wire industry, the most important section is multi wire / bunching section of the production since most of the challenging conditions and constraints are valid in this section and it is possible to deploy the production plan to other sections easily if the scheduling plan of the section is good enough. In this study, we will define the properties of the problem and the objective function within a mathematical model. After this, we will demonstrate a scheduling heuristic to cover both the multi-wire and bunching stages of this section. In multi-wire section, after acquiring an initial solution, with simulated annealing technique, final solution will be obtained. This final solution of the first stage is given as an input to the second stage, bunching, and jobs in bunching stage are scheduled with another heuristic and final result is revealed out. The final result will be compared with the planners schedule on the system and efficiency of the proposed heuristic methods will be evaluated. Heuristic algorithms in both multi-wire and bunching stages considers the relevant constraints like tardiness to the due date, sequence dependent setup times, machine eligibility constraints, parallel machines in bunching, spool weight and type. Tests about the heuristic are performed with the real production data of the Turkey's and regions one of the biggest copper-wire production company, "Erbakır Elektrolitik Bakır Mamulleri A.Ş.", located in Denizli-Turkey.

2. PROBLEM DEFINITION

Production of copper has several stages and after each stage, semi-finished products are produced and stored in stocks. Each stage has different constraints and different alternative machine sets located in three main physical locations (L_1, L_2, L_3). Summary of production stages and their properties can be found in Table 2.1. Also Figure 2.1. describes the general flow of the production according to the different stages. Decomposition of the production is also shown in Figure 2.1.

Table 2.1. All production sections and properties of each section

PRODUCTION LEVEL	PROPERTY
Rod Breakdown	• Non-identical alternative parallel machines in terms of output type, • Production is based on forecast and customer order. Push production is the main production type, • Machines are located in Location 1 and Location 3.
Tin Plating	• Identical alternative parallel machines, • Machines are located in Location 1 and Location 3.
Intermediate Wire Drawing	• Non-identical alternative parallel machines in terms of spool type, • Machines are located in Location 1, • Production speed of the machines (i.e. production output per time) can be adjusted when necessary.
Fine Wire Drawing	• Identical alternative parallel machines • Machines are located in Location 1, • Machines are not eligible to produce all products. Some of them are for only special

	products.
Multi-wire Drawing	• Non-identical alternative parallel machines • Machines are located in Location 1, Location 2 and Location 3, • Special constraints exist according to product and order. These are discussed in chapter 2.2.
Thin Bunched Wire	• Non-identical alternative parallel machines • Machines are located in Location 1, • Special constraints exist according to product and order. These are discussed in chapter 2.2.
Thick Bunched Wire	• Non-identical alternative parallel machines • Machines are located in Location 1, Location 2 and Location 3, • Special constraints exist according to product and order. These are discussed in chapter 2.2.
Transformation	• Identical alternative parallel machines, • Machines are located in Location 1.
Rolling	• Identical alternative parallel machines, • Machines are located in Location 1.
Packaging	• Hand packaging, • No capacity limit.

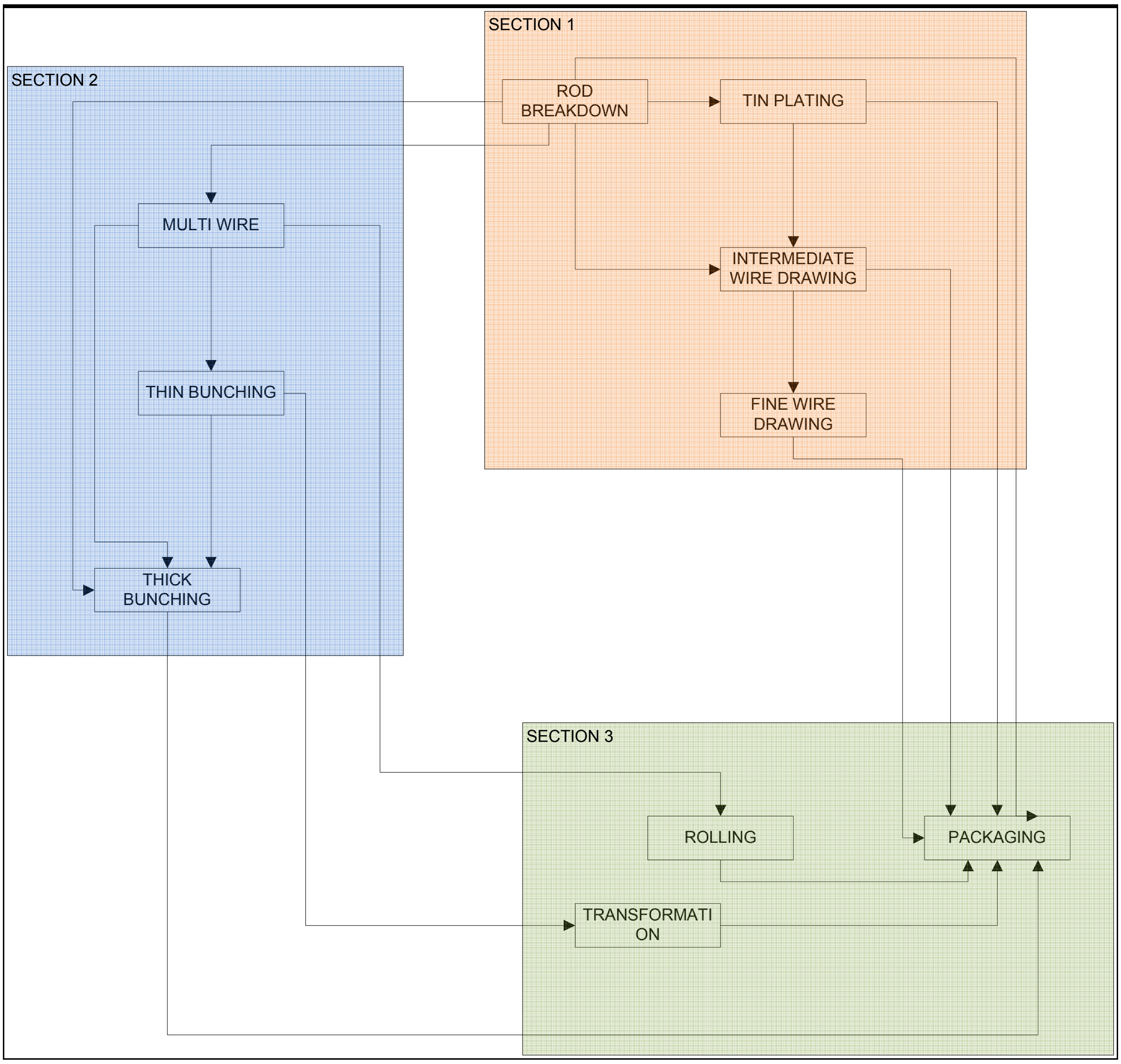

Figure 2.1. Production stages and sections of the company

The total production planning problem is very complex and it is possible to divide the process flow into three logical sections. The first section is composed of rod breakdown, tin plating, intermediate wire drawing and fine drawing stages. The stages in this section have few alternative machines relative to the other stages and the number of complex constraints is considerably less. Planning in this section is usually based on forecast and the production type is push-production. Similar situation is also valid in section 3, which is composed of rolling, transformation and packaging stages. In those stages, there are also few machine alternatives and the capacity of these machines are high enough to cover the production load coming from previous section, section 2 and the stages in this section can be identified as non-bottleneck. However, in section 2, where multi-wire and bunched wire (both thin and thick) are produced, the number of planning related issues and properties are higher when compared to other sections. Having a good production plan in this section brings a good overall production plan. In this study, we will focus on the production in multi-wire drawing, thin bunching and thick bunching stages. For the sake of simplicity, bunching stages are combined and the stages are named as “Stage 1“ and “Stage 2” where “Stage 1” is multi-wire and “Stage 2” is bunched wire stages.

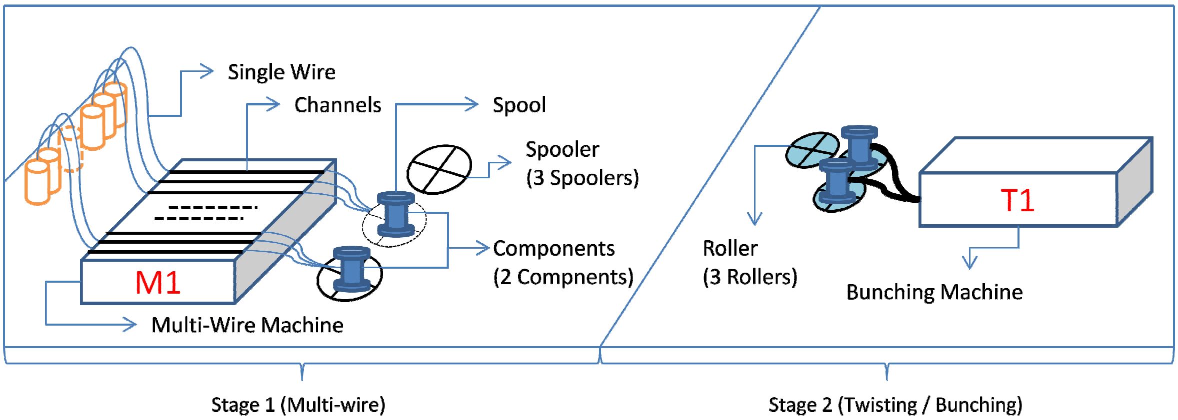

Figure 2.2. Problem specific terminology

2.1. Business Notations

During describing the properties of the problem, abbreviations are used for some specific scheduling related business parameters. These are defined in this section.

2.1.1. Job Notation

Jobs have many planning related attributes. In our study, jobs are defined with the following notation in the company:

WC_i : Bunched wire count of job i

r_i : radius of job i

TP_i : Tinned or plane indicator of job i

CS_i : Spool count of the customer requirement of job i

SW_i : Single customer spool weight of job i

d_i : due date of job i

As an example, 19 / 0,25 / P / 5 / 150 / 20100325 stands for a sales order of 5 spools 150 kg each for date 23.03.2010 that consist of 19 bunched plane wire with 0.25 mm dimension of each single wire.

2.1.2. Other Notations

Apart from the jobs, planning related other parameters are also needed to be shown during representation of the problem. There parameters are as follows:

L_j^s : Location of machine j at stage s
v_j^1 : Spool count of machine j in multi − wire stage
v_j^2 : Roller Count of machine j in bunched wire stage
SC_k^{min} : Minimum weight limit of spool type k
SC_k^{max} : Maximum weight limit of spool type k
tw_i : Total weight of job i

2.2. Properties of the Problem

Properties of the scheduling problem about the selected two stages are described in this section.

- Property 1 states that there are non-identical alternative machines in both Multi-Wire and Bunched-Wire production. Machines in these stages have several alternatives. However, these alternatives are not unique and eligibility constraints exist according to several criteria. One of them is production speed. Production speed depends on the following factors: Single wire dimension, tinning property of the production, spool type. Using these factors, production speed can be calculated for each machine. The other is products that are suitable to be produced. It is not possible to produce every product in every alternative machine. For example, products having smaller dimension cannot be produced on some of the multi-wire and bunching machines. Input and output spool form is another criterion. Each machine has a set of acceptable input and output spool form in both stages. Machine assignment affects the spool type selection. Input and output roller counts are also important Bunched wire machines have different input and output roller counts. In bunched wire production, input rollers determine how many multi-wire spools can be bunched together. This constraint, limits the alternative in machines and semi finished products. Channel count is the final criteria. In multi-wire production, machines are consisting of channels and this differs from alternative to alternative. Channel number is the limiting factor according to how many single wires will be brought together and a multi wire will be produced.

- Property 2 states that customer orders cannot be divided into separate multi wire machines. One job in multi-wire production stage must only be produced in one machine. In another word, if a machine is selected for a job in stage 1, other alternative machines cannot be used for this job in parallel.
- Property 3 states that bunching operations can be divided into parallel machines. A bunching job can be produced in alternative parallel bunching machines simultaneously. For example, there is a sales order as follows: 19 / 0,25 / P / 5 / 150 / 20090725. This sales order requests 5 spools and these spools can be produced at the same time in 5 different bunching machines, if spool sizes allows. However, customer order spool number is a limiting factor here. Parallel machines cannot exceed total spool number of the sales order.
- Property 4 states that there are alternative component counts for each job. Same bunched wire can be produced by using different multi-wire combinations. Company statistics shows that on average one bunched wire may have at most three alternatives and 50% of them have only one alternative. For example, it is possible to produce a job of 19 bunched wire (19 / 0,25 / P) with two alternatives. One is (10 + 9) multi wires – 2 components. The other is (6 + 6 + 7) multi wires – 3 components. During planning, correct alternative should be selected according to setup, due date and spool weight constraints. Machine selection in stage 1, multi-wire, determines the component count (i.e. If a job is assigned to a machine, there is only one alternative of production and the number of components is known in advance).
- Property 5 states that spooler count of multi-wire machine and roller count of bunching machine affects machine selection in both stages. Each alternative machine in multi-wire stage and bunched wire stage has different spooler (v_j^1) and roller (v_j^2) counts. During job assignment in production planning three scenarios may happen.

The first one is rollers of a machine can be fully used. The second one is some of rollers of a machine may be left unused. The third one is additional portable rollers are added to a machine. In ideal case, rollers should be fully utilized and alternative machine selection should be done according to the roller counts.

- Property 6 states the spool type compatibility of the alternative machines in multi-wire and bunched wire stages. Semi finished products after stage 1, multi-wire, are carried to the second stage, bunched wire, on different types of spools. As stated in Property 1, some of the machines in both stages may accept some of the different types of spools. It is known beforehand whether a machine may accept a spoon type or not.
- Property 7 states that there is a minimum and maximum spool size between stages according to the spool type. During transfer of products from stage 1 to stage 2 different spool types can be used, which is also declared in Property 6. Each spool type has a minimum and maximum capacity that it can carry on. In another words, it is not possible to carry more than and less than a certain amount of weight on a specific spool type between stages. Let's consider the job 19 / 0,25 / P / 5 / 300 / 20090725. Each customer spool in this job is 300 kg and 5 spools are requested. Total weight of the job is 1500 kg (tw_i). Number of parallel machines in stage 2 and component count in stage 1 affects the spool type, therefore machine selection. Two alternatives are possible for production. Table 2.3 and 2.4 shows these alternatives. Single spool size can be found with the formulation in Figure 2.3. On the other hand, Figure 2.4. illustrates an example of component count and spooler count compatibility. In the first part, production is in machine M1 at stage 1. This product is produced with two components in M1, which is predefined. On the other hand, multi-wire is transferred to the second stage on blue spools due to the weight constraints. At stage 2 there are three alternative machines. The first and second machine accepts blue spools and the spooler count is greater than or equal to the component count, which is two. However, third machine does not accept blue spool type and this machine is eliminated among the alternatives for this job. In the second part, production is on machine M2 in stage 1. This product is produced with three components in M2, which is predefined. On the other hand, multi-wire is transferred to the second stage on green spools due to the weight constraints. At stage 2 there are three alternative machines. The second and third machines accept green spools but

the spooler count of M2 is two, which is less than component count and this machine becomes ineligible for this job. Also, first machine does not accept green spool type and this machine is eliminated among the alternatives for this job.

Table 2.2. Spool type limits

Spool Type 560	SC_k^{min} = 70 ; SC_k^{max} = 200 Kg
Spool Type 630	SC_k^{min} = 250 ; SC_k^{max} = 450 Kg
Spool Type 800	SC_k^{min} = 800 ; SC_k^{max} = 1000 Kg

Table 2.3. Spool size constraint - (10 + 9) multi-wires – 2 components.

Parallel machines in bunched wire	**Component count**	**Single spool weight**	**Possible spool types**
1	2	1500 / 2 / 1 = 750	None
2	2	1500 / 2 / 2 = 375	630
3	2	1500 / 2 / 3 = 250	630
4	2	1500 / 2 / 4 = 187,5	560
5	2	1500 / 2 / 5 = 150	560

Table 2.4. Spool size constraint – (6 + 6 + 7) multi-wires – 3 components.

Parallel machines in bunched wire	**Component count**	**Single spool weight**	**Possible spool types**
1	3	1500 / 3 / 1 = 500	None
2	3	1500 / 3 / 2 = 250	630
3	3	1500 / 3 / 3 = 166,7	560
4	3	1500 / 3 / 4 = 125	560
5	3	1500 / 3 / 5 = 100	560

Single Spool Size = Total Weight of the Job /
Component Count /
Parallel machines in stage 2

Figure 2.3. Single spool size formulation

- Property 8 states that due dates are soft constraints. Due dates should be obeyed and tardiness should be minimized during planning.

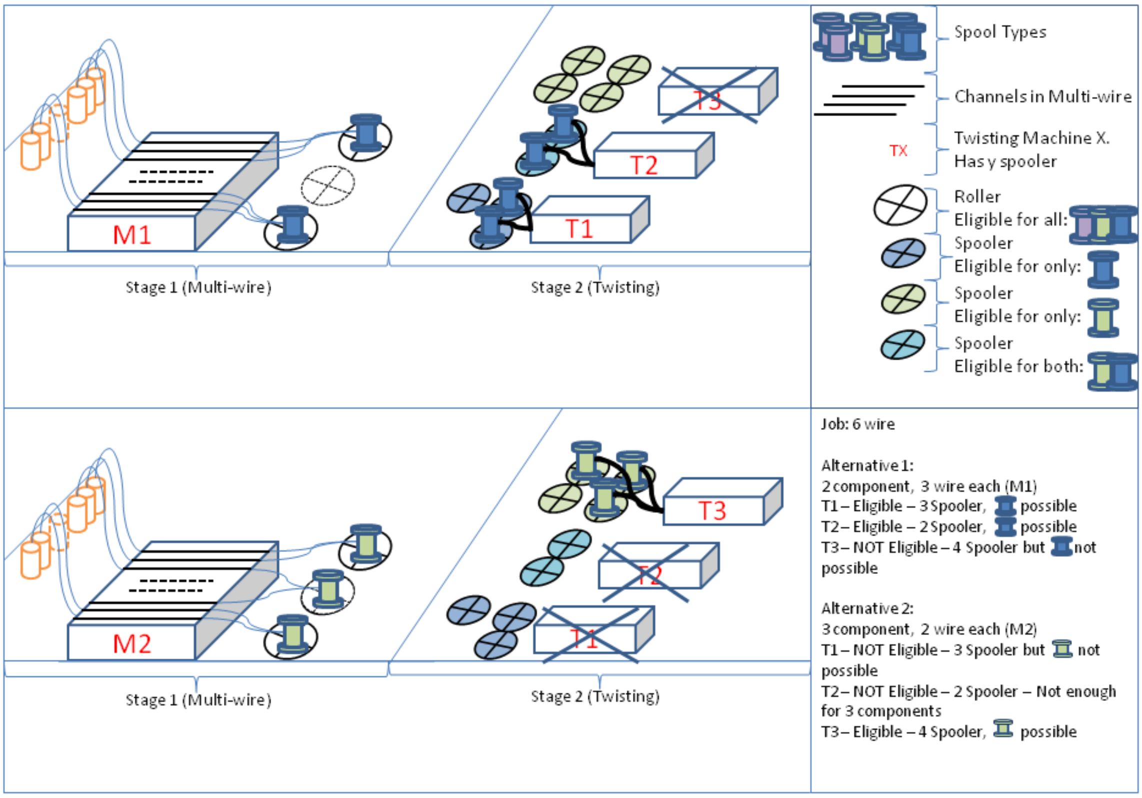

Figure 2.4. Component / roller count illustration

- Property 9 states that holding WIP inventory is not desirable. Since the produced material is very expensive, inventory holding is an important issue in copper manufacturing. Business requires keeping lowest amount of WIP and during scheduling, inventory holding costs should also be controlled.
- Property 10 defines adjustable production speeds on machines. Multi-wire machines have two different production speeds. Each machine can work in fast mode and in normal mode at a time. This is the characteristic of the machine and this characteristic changes occasionally (5 times in one year for all machines on average). This affects the unit production quantity.
- Property 11 defines setup sequencing in multi wire production. Setup sequencing is critical in multi-wire production. Radiuses of the single wires should be close to each other in order to get the minimum setup time and cost.

- Property 12 defines setup sequencing in bunching production. Sequencing is not critical in bunching machines. Setup duration can be accepted as fixed duration or can be ignored.
- Property 13 states that physical location of multi-wire and bunched wire machines. Production machines of different levels are located to different physical locations (L_j^s). There are three different physical locations (L_1, L_2, L_3) and in each location it is possible to produce several levels of the production, including multi-wire and bunched wire. In both stages, there are machines which are alternative to each other and located in different locations. If a lob is assigned to a machine in multi-wire production having different location than the machine assignment in bunched wire stage, then spools should be carried from one location to another, which is not desired. Therefore, spool movements between locations should be minimized.
- Property 14 states that multi wire and bunched wire jobs overlaps. Bunched wire production can start after one set of spools production is finished in multi wire level. Therefore, scheduling of these operations may overlap. This overlap quantity depends on the production alternative in multi wire and machine selection in bunched wire levels.

2.3. Assumptions of the Problem

During mathematical representation and definition of solution procedure of the problem, several assumptions are made. These assumptions are as follows.

- Assumption 1 states that the problem is divided into levels. As stated at the introduction section, production planning problem of this copper wire production company can be divided into three sections. In this study, we will concentrate on the production planning of the middle section, which is composed of multi-wire and bunched wire stages. First and third sections are assumed to be given and they will not be considered during planning procedures.
- Assumption 2 suggests that production quantities and WIP stocks are accepted as given. Production quantities and WIP stocks are accepted as given for stage 1.
- Assumption 3 defines overlapping of jobs between stages. In property 14, overlapping condition between multi-wire and bunched wire stages are explained.

However in some cases moving spools one by one from stage 1 to stage 2 may not be desirable and spools are transferred to the second stage when all the items of the job is finished in stage 1. Therefore during planning, overlapping of stages is not considered in this study.

- Assumption 4 states that one job has only one unit production duration in one machine. In property 10, it is explained that machines may have variable production duration for a single job according to the speed adjustment on the machine. Speeds of the machines are rarely adjusted and at the start of the planning it is assumed that production speed and production duration of each job is fixed and given as input parameter.

3. LITERATURE REVIEW

In this section, several academic papers related to our scheduling problem are reviewed. The review of the literature initiated with the general studies on flowshop problems. Taxonomies of the problem are analyzed to gain overall insight and parallel machine problems in literature are investigated. Since the machines in our problem are unrelated and demonstrate different properties for a single job, a section is dedicated to the review of unrelated parallel machine problems. Papers dealing with setup costs and times in flowshop environment, proposing new solution techniques and considers the batching of jobs are also summarized. Finally, a special section is dedicated for the review simulated annealing method, which we will be using in our heuristic.

A hybrid flowshop (HFS), as described by Linn and Zhang [1], consists of a series of production stages, each of which has several machines operating in parallel. Some stages may have only one machine, but at least one stage must have multiple machines. The flow of jobs through the shop is unidirectional. Each job is processed by one machine in each stage, and it must process through one or more stages. The relationship of the machines in each stage can be identical, uniform, or unrelated.

In literature there are many researches on hybrid flow shops and the number of researches in this areas is increasing year by year due to the fact that flexible flow lines can be found in vast number of industries including automotive, chemical, cosmetics, electronics, food, packaging, paper, pharmaceutical, printing, textile and wood processing [2] The implementations on those sectors are investigated in various researches.

Quant and Khun [3] gave a detailed taxonomy of flexible flow line problems in literature until 2004. They first categorize the solutions to the flexible flow line problem as optimal and heuristic approaches. Heuristic procedures may be segmented into holistic and decomposition approaches. Decomposition approaches divide the problem with respect to the individual jobs, the production stages, or the sub-problems as batching, loading, and sequencing. This taxonomy explains the previous researches in each sub problem and gives the brief summary of the solution approaches in every category. They concluded that the integration between flexible flow line scheduling and higher level planning phases is the area that can be researched for further studies. Also Gupta and Stafford [4] have an extensive research about the evaluation of the flowshop scheduling problems starting from 1954 up to 2004. They give a clear categorization of the problem and presents the solution procedures. They also state that the mathematical theory of flowshop scheduling suffers from too much abstraction and too little application and the practical use of flowshop scheduling techniques, therefore, is rare. Future research in flowshop scheduling should be inspired more by real life problems rather than problems encountered in mathematical abstractions.

Parallel machine problems are divided into two sections: Uniform parallel and non-uniform parallel. Koulamas and Kyparisis [5] investigated the existing LPT algorithm that minimizes the makespan and improved it by sequencing the longest three jobs optimally. Their results demonstrate the applicability of this approach (already implemented for identical parallel machine scheduling problems) to a uniform parallel machine environment. Apart from makespan minimization, in literature there are also researches about uniform parallel machines that deals with the objectives related to tardiness. Yalaoui and Chu [6] handles the identical parallel machine scheduling problem to minimize total tardiness in their research. They aim at developing an exact method to solve this problem. Their computational experiments show that they obtain optimal solutions in some cases with 30 jobs and 2 machines within a reasonable amount of computation time. Line and Jeng [7] give a solution approach to the problem of minimize the maximum lateness and the number of tardy jobs. In this paper, they address a batch scheduling problem where a set of identical parallel machines is available for processing the jobs in batches in the continuous mode. Two dynamic programming algorithms were proposed for finding optimal solutions to the two objectives, respectively. The proposed algorithms take exponential times in delivering optimal schedules. The computational complexities become pseudo-polynomial when the number of machines is fixed, where in a realistic production environment, the number of machines is usually quite limited and fixed. One other research on minimizing the total tardiness in unrelated parallel machines belongs to Armentano and Filho [8]. Their paper explains a solution to the problem of scheduling jobs in uniform parallel machines with sequence-dependent setup times in order to minimize the total tardiness (PSTP Problem). They propose versions of the greedy random search adaptive procedure (GRASP) that incorporate adaptive memory-based approaches for solving the PSTP.

However, in real word applications, mostly machines are not uniform; rather they are unrelated to each other. In recent years, there are many researches in this field since the researchers are keener on the applicability of the solution procedures to the industrial real problems. For this purpose, Ruiz and Maroto [9] proposed a genetic algorithm that takes sequence dependent setups and machine eligibility into account for ceramic tile industry. A computational experiment has been conducted to check the performance of the proposed algorithm. The statistically significant results indicate that the proposed algorithm is between 53% and 135% better than the second best method. Chen and Chen [10] also examined a similar problem but the objective is to minimize the total tardiness. They developed a bottleneck based heuristic for the scheduling problem. Their heuristics first identify the bottleneck stage in the flow line, and then separate the flow line into the upstream stages, the bottleneck stage, and the downstream stages. Their computational results show that the bottleneck-based heuristics significantly outperform all the dispatching rules for the test problems. They extended their research with a new article in 2009 [11] where the objective function is not minimizing the total tardiness, rather minimizing the makespan. They concluded that their heuristic outperforms the well known heuristics in literature. Bellanger and Oulamara [12] consider a two-stage hybrid flowshop scheduling problem, where each of n tasks is to be processed first at stage one and then at stage two. The first stage contains several identical discrete machines and the second stage contains several identical batching machines where they considered the makespan criteria. Their research is motivated by the scheduling of tire in the manufacturing industry. There are also other industrial solutions researches about unrelated parallel machine problems. Yaurima et al. [13] present a genetic algorithm for the production environment of a television assembly line and focused on suboptimal scheduling solutions for the hybrid flowshop with unrelated machines, sequence-dependent setup time, availability constraints, and limited buffers. Their experiments show that the proposed algorithm shows an improvement in scheduling. They extend the study of Allaoui and Artiba [14]. The research of Low et al. [15] aims to fill the gap in the literature that takes both functional constraints and unrelated alternative machines cases into account simultaneously. This problem occurs in certain practical manufacturing environments such as semiconductors, electronics manufacturing, airplane engine production, and petrochemical production. Hence, the purpose of this report is to propose some efficient heuristics for solving this problem. The computational results demonstrate the effectiveness of the heuristics.

One other important characteristic of the flowshop scheduling problems is the setup between jobs. Setup is ignored by some of the papers but the other researches about hybrid flowshop considers setup either as sequence dependent or sequence independent. The first comprehensive survey paper on scheduling problems with separate setup times or costs were conducted by Allahverdi et al. [16]. They reviewed the literature and surveyed 190 papers between the mid-1960s to 1999. Allahverdi et al. [17] extended this survey from 1999 to 2006. They surveyed more than 300 papers on scheduling with setup times (costs). When compared with the Allahverdi et al. [16], it is clear that the number of studies about setup times and costs is increasing dramatically year by year. The majority of the papers addressed sequence-independent setup times because dealing with sequence-dependent setup times is more difficult.

Allahverdi and Al-Anzi [18], when addressing a two stage assembly scheduling problem to minimize the makespan, they considered the sequence independent setup times and proposed three heuristics in his research. They concluded that a new version of self-adaptive differential evolution outperforms other heuristics. Ferreira et al. [19] also consider the setup times and costs but unlike Allahverdi and Al-Anzi's study [18], they took sequence dependent setup times and costs. Their study is based on a real life problem in soft drink industry and they proposed a mixed integer programming model that considers the production bottleneck alternates between liquid flavour preparation and bottling stages. The results show that the solution approaches are capable of producing better solutions than those used by the company. One other real word study on sequence dependent setup times belongs to Voβ and Witt [20]. They aim to minimize the weighted tardiness and considered a 16 stage production problem in German steel sector. They extended the well known resource constrained project scheduling problem by adding sequence dependent setup costs. Naderi et al. [21] proposed two algorithms that deal with the flexible flow shops and setup characteristics of this problem. Algorithms are compared against seven other existing algorithms and they are very competitive for the studied problem statistically. Zandieh and Jabbari [22] added machine availability constraint to the hybrid flowshop with sequence dependent setup scheduling problem, which is a case in real life problems. Their algorithm minimizes the makespan and outperforms well known heuristics.

Also in literature there are researches that investigate the special solution procedures and constraints to the problem. One of them is summarizing the papers about TSP based approaches for flowshop scheduling [23]. In this paper, the relationship between the traveling salesman and the flowshop scheduling problems is investigated. Several practical applications of TSP based approaches to flowshop problems are reviewed and first detailed summary of the results in this area are presented. Nishi et al. [24] presented a new Lagrangian relaxation for solving the hybrid flowshop scheduling problem to minimize the total weighted tardiness. Zhenbo and Wenxun's paper [25] describes a new variant of the parallel machine scheduling problem with off-line and on-line scheduling. Amin-Naseri and Beheshti-Nia [26] studied the hybrid flowshop problem on parallel batching perspective. In parallel batching it is assumed that machines in some stages are able to process a number of operations simultaneously. They developed three heuristic algorithms to give near optimal solutions. Laub et al. [27] added multiple orders per job constraint to the flowshop problem and an optimization model is presented that addresses both job formation and job sequencing. They define a heuristic to minimize the makespan for the problem for two-machine item-processing flowshops. Most of the researches in literature take into consideration as an objective functions about either makespan or tardiness. There are very few research paper that takes both of the objective function into account. Toktaş et. al. [28] address the problem of minimizing makespan and maximum earliness simultaneously in a two machine flow shop environment. They propose a heuristic procedure that generates approximate efficient solutions.

Consider a scenario where lots consisting of several discrete and identical items are to be processed on several machines configured as a flow shop. Instead of transferring the entire lot after all of its items have been processed on a machine, consider transferring the items of the lot in smaller batches called sublots. This technique of splitting a lot into sublots (also termed transfer lots), and processing different sublots simultaneously over different machines, albeit still maintaining their movement over the machines in accordance with their flow shop configuration, is called lot streaming. [29] There are plenty of research papers on lot streaming problems. It is possible to categorize the problem as two-machine, three-machine, and the general m-machine flow shop problems [29]. Sarin and Jaiprakash's study gives details of this classification and identifies the researches that address each category. Langevin et al. [30] present methods to determine optimal transfer batch sizes of various part types between two machines to minimize the sum of the total relevant costs. Sriskandarajah and Wagneur [31] provide algorithms to simultaneously deal with the lot streaming and scheduling problem of multiple products in two-machine no-wait flowshops to minimize makespan. Chen and Steiner [32] study the structural properties of schedules which minimize the makespan for a single job with detached setup times in a flow shop. They performed their studies in three-machine flow shop environment. Setups are also an important factor in lot streaming problems. Kalir and Sarin [33] presented a near optimal solution for the flowshop lot streaming problem with sublot-attached setups. A fast, optimal solution algorithm for the single batch problem is presented. For the multiple batch problem, they propose a near optimal solution procedure which is optimal in two-machine flow-shops. Huang and Yang [34] approached the problem with an Ant Colony Optimization solution. They proposed a simple and effective search model for solving the problem of overlapping production schedule planning with multiple objectives using ant colony optimization (ACO). Martin [35] presents a hybrid genetic algorithm/mathematical programming heuristic for the n-job, m-machine flowshop problems with lot streaming. As a new aspect to this kind of problems, the interleaving of sublots from different jobs in the processing sequence, is developed and addressed

There are well-known solution techniques for the flowshop problems like genetic algorithm, tabu search, simulated annealing etc. Logendran et al. [36] proposes a methodology for minimizing the weighted tardiness of jobs in unrelated parallel machining scheduling with sequence-dependent setups. They presented several heuristic for different sized problems and used tabu search technique. They make recommendation of tabu list sizes according to the size of the problem. Kim et al. [37] presents a scheduling problem for unrelated parallel machines with sequence-dependent setup times, using simulated annealing. They determined a scheduling policy so as to minimize the total tardiness. They presented several neighborhood selection methods and compared them to identify the method that outperforms the others. Lee and Pinedo [38] present a three stage heuristic for a identical parallel machine with setup times problem. They used simulated annealing method as a post-processing procedure and the performance of their heuristic is measured for different parameters. Weng et al. [39] addresses the problem of scheduling a set of independent jobs on unrelated parallel machines with job sequence dependent setup times so as to minimize a weighted mean completion time. In their study, seven heuristic algorithms are proposed and tested by simulation. The results are reported and discussed.

Although in literature, most of the aspects of our problem are investigated individually, there is no specific review that covers all the constraints of the properties of this scheduling problem. Also, there is a lack of literature that covers the scheduling problems in copper and wire industry. In this master thesis, we will fulfill this gap and try to present a solution procedure that covers most of the properties of the problem.

4. MATHEMATICAL FORMULATION OF THE PROBLEM

The flexible flow line (FFL) manufacturing system considered in this thesis assumes that there are two stages of production. There are unrelated parallel machines in each stage and the number m^s may vary from stage to stage. There are N jobs to be processed and each job must visit both stages consecutively. The processing time of a job on a stage is dependent on the machine in the stage assigned to the job, and it is known in advance. A machine can process only one job at a time, and jobs cannot be preempted. Each job should start and be finished at only one machine in stage 1. However in stage 2 each job can be processed on p_i parallel machines at a time. There are unlimited buffers between stages, and there is no machine breakdown before jobs are processed on any machine. Machines in stage 1 and stage 2 have different properties. Machines have different number of roller at stage 1 and roller at stage 2 ($v_j^2$and r_j^1). For job i it is also possible to have different alternative component counts (b_i) and this is a decision variable of our mathematical model. Machine assignment in stage 1 decides the component count. There are sequence dependent setup costs between jobs at stage 1. Setup costs are measured as the difference between the radius of job i (r_i) and the average radius on that machine ($\overline{r_j}$). Setup costs in stage 2 are ignored. Also, we assume that all the jobs and the machines are available at time zero.

4.1. Objective Function

The aim of the scheduling problem considered in this study is to minimize the sum of setup costs in stage 1, tardiness of all jobs and location changes between stages. (Property 8) Tardiness is calculated as the sum of the difference between due date and assignment period *t* of job.

$$\sum_i T_i \tag{4.1}$$

In copper-wire industry, setup cost of a job is measured as the difference between the radius of the job *i* and the radius of previous job *(i-1)* on machine *j* (Property 11). Since we consider our problem as allocation of jobs on machines and time periods, setup cost is measured in this study as the difference of maximum radius and the minimum radius that is assigned to a machine *j* at period *t*.

$$\sum_{t}\sum_{j}(r_{jt}^{max} - r_{jt}^{min}) \tag{4.2}$$

There are three physical locations on the shop floor. Machines in stage 1 and stage 2 are located in these locations. If a job is assigned to a machine in a specific location in stage 1, and the same job is not assigned to a machine which is in the same location in stage 2 than this job should have a penalty and the objective is to minimize these penalties (Property 13).

$$\sum_{i} dl_i \tag{4.3}$$

The objective function is defined by weighted sum of setup cost, tardiness and location change cost as follows:

Minimize:

$$MW1 \times \sum_{i} T_i + MW2 \times \sum_{t}\sum_{j}\left(r_{jt}^{max} - r_{jt}^{min}\right) + MW3 \times \sum_{i} dl_i \tag{4.4}$$

4.2. Constraints

The sum of machine assignment index of a job *i* over each machine in stage 1 and period should be equal to 1 (Property 2). Eq. (4.5) shows that one job *i* can only be assigned to only one machine *j* and one period *t*.

$$\sum_{t}\sum_{j} x_{ijt}^{1} = 1\,, \qquad \forall\, i \tag{4.5}$$

In stage 2, it is possible to assign jobs to p_i parallel machines simultaneously. Therefore, Eq. (4.6) guarantees that one job at stage 2 can be assigned to only p_i parallel machines. p_i is a decision variable of the problem and its value is an output of the solution (Property 3). However, the parallel count cannot exceed the customer spool number in the sales order. Eq. (4.7) gives this limitation.

$$\sum_{t}\sum_{j} x_{ijt}^2 = p_i , \qquad \forall\, i \tag{4.6}$$

$$p_i \le CS_i, \qquad \forall\, i \tag{4.7}$$

After processing in stage 1, spools are transferred to stage 2 and are put on rollers. Each machine in stage 2 has different number roller v_j^2 and the component count decision that is given in stage 1 should not exceed the number of roller of the machine in stage 2 (Property 4 and Property 5) Eq. (4.8) represents this condition such that, if a job *i* is assigned to machine *j* at stage 2, the component count b_i should be smaller or equal to the roller count of machine *j* v_j^2.

$$M\left(\sum_{t} x_{ijt}^2 - 1\right) + b_i \le v_j^2, \qquad \forall\, i, j \tag{4.8}$$

In this equation, if the job *i* is assigned to a specific machine j', the left side of the equation will be:

$$M(1-1) + b_i \rightarrow b_i \le v_j^2$$

If the job *i* is not assigned to machine j', then the left side will be:

$$M(0-1) + b_i \rightarrow -M + b_i \le v_j^2$$

which is always satisfied and this constraint will become redundant for this particular machine j'.

Each job can be produced with alternative components counts in stage 1. Machine assignment in stage 1 decides the component count. Eq.(4.9) represents the equation between decision variable b_i, and x_{ijt}^1 and parameter B_{ij}^1.

$$b_i = \sum_t \sum_j x_{ijt}^1 \times B_{ij}^1 , \qquad \forall\, i \tag{4.9}$$

Between stage 1 and stage 2, components are carried on *k* different type of spools. In stage 1, each machine j is known before whether it is eligible to produce spool type k which is shown as a_{kj}^s. On the other hand, likewise stage 1, in stage 2 some of the machines can use certain spool types and some cannot (Property 6).

$$\sum_t x_{ijt}^s + S_{ik}^{1-2} \leq 1 + a_{kj}^s, \qquad \forall\, i, j, k \tag{4.10}$$

In Eq. (4.10), if a specific job i' is assigned to machine j' at time t', the equation is:

$$1 + S_{ik}^{1-2} \leq 1 + a_{kj}^s \rightarrow S_{ik}^{1-2} \leq a_{kj}^s .$$

This shows that if it is ineligible to assign spool type k to machine j, (i.e., $a_{kj}^s = 0$), it is not possible to transfer on spool k between stage 1 and stage 2 (i.e., S_{ik}^{1-2} must be 0). If it is eligible to assign spool type k to machine j, (i.e., $a_{kj}^s = 1$), it is possible to transfer on spool k between stage 1 and stage 2 (i.e S_{ik}^{1-2} can be 0 or 1).

If job i' is not assigned to machine j' at time t', the equation is:

$$0 + S_{ik}^{1-2} \leq 1 + a_{kj}^s \rightarrow S_{ik}^{1-2} \leq 1 + a_{kj}^s .$$

which is always satisfied.

Spool type selection between two stages considering machine eligibility is defined in Eq. (4.10). Eq. (4.11) and Eq. (4.12) defines the spool type selection by considering the weight limitations of the spool types.

Each spool type can accept wire between certain weight limits. In another word, it is not possible to roll more than SC_k^{max} and less than SC_k^{min} kilogram of wire on spool type k (Property 7). The weight of job i is one of the parameters of the model (tw_i) and the weight of each spool is calculated as

$$\frac{\text{total weight of job i}}{\text{parallel number of machines in stage 2} \times \text{number of components}}$$

This value should be greater than the minimum spool limit of type k if this job i is assigned to spool type k. Eq. (4.12) deals with the maximum weight limit of spool types. The weight of each spool between two stages should be less than the maximum spool weight limit of spool type k.

Eq. (4.11) and Eq. (4.12) are non-linear constraints since several decision variables are multiplied in these inequalities.

$$M(S_{ik}^{1-2} - 1) + SC_k^{min} \leq \frac{tw_i}{p_i \times b_i}, \qquad \forall\, i, k \tag{4.11}$$

$$M(1 - S_{ik}^{1-2}) + \; SC_k^{max} \;\geq \frac{tw_i}{p_i \times b_i}, \qquad \forall\, i, k \tag{4.12}$$

Spools between stage 1 and stage 2 are transferred on just only one type of spool k. Eq. (4.13) guarantees that only one type of spool is assigned to job i between stages.

$$\sum_k S_{ik}^{1-2} = 1\,, \qquad \forall\, i \tag{4.13}$$

Machines have defined capacities for each period. Capacity load on machine j, which is calculated by summing up the duration of processing time of all jobs on machine j, in period t should not exceed the maximum available capacity of the machine j in period t. This balance is shown via Eq. (4.14).

$$\sum_i (x_{ijt}^s \times u_{ij}^s) \leq C_{jt}^s, \qquad \forall\, j, t \tag{4.14}$$

If a job is assigned to a time period which is greater than the due date, this job is considered as tardy. Eq. (4.15) sets a tardiness bound for each job. If a job is assigned to a period t', then x_{ijt}^2 parameter will be 1 and the equation becomes $T_i \geq t - d_i$. If this time period is later than the due date (i.e., $t > d_i$), tardiness of this job should be greater than or equal to the difference (i.e., $T_i \geq t - d_i > 0$). If the assigned time period is earlier than the due date (i.e., $t < d_i$), this constraint is redundant since then it becomes negative number, which is always satisfied (i.e. $T_i \geq t - d_i < 0$). Similar scenario occurs if the job is not assigned to this time period, this time x_{ijt}^2 is 0 and the equation is $T_i \geq -d_i$ which is again always satisfied.

$$T_i \geq t \sum_j x_{ijt}^2 - d_i, \qquad \forall\, i, t \tag{4.15}$$

Production in stage 1 must always be done before the production in stage 2 for all jobs. To represent this, Eq. (4.16) is added to the model. In Eq. (4.16) if a job is assigned to period t' in stage *1*, the right side of the equation should be equal to 1, which means this job should be assigned to any machine after period t at stage 2 (i.e., $\sum_j \sum_{y \geq t} x_{ijy}^2 \geq 1$). If the job is not assigned to a machine at period t', this constraint is always satisfied and becomes redundant (i.e., $\sum_j \sum_{y \geq t} x_{ijy}^2 \geq 1 - M$).

$$\sum_j \sum_{y \geq t} x_{ijy}^2 \geq 1 - M(1 - \sum_j x_{ijt}^1), \qquad \forall\, i \tag{4.16}$$

The objective function aims to minimize the setup time within the periods. For this, maximum and minimum radiuses of jobs that are assigned to the machines are used in the objective function. Maximum radius is found by forcing the r_{jt}^{max} variable to be always greater than or equal to the radiuses that are assigned to a specific machine. Eq. (4.17) shows this relation. On the other hand, for minimum radius parameter, Eq. (4.18) forces parameter r_{jt}^{min} to be smaller than or equal to the radiuses that are assigned to a specific machine. If there is no assignment to this machine, this equation is always satisfied (Property 11).

$$r_{jt}^{max} \geq r_i x_{ijt}^1, \qquad \forall\, i, j, t \tag{4.17}$$

$$r_{jt}^{min} \leq r_i + M\left(1 - x_{ijt}^1\right), \qquad \forall\, i, j, t \tag{4.18}$$

Location differences are critical factor in copper production. Using machines at the same location for both of the stages are highly preferred and movement between locations are tried to be minimized. Decision variable dl_i indicated the location differences of the jobs and in the objective function this parameter is set to be minimized (Property 13).

$$dl_i = \sum_t \sum_j x_{ijt}^1 \times L_j^1 - \sum_t \sum_j x_{ijt}^2 \times L_j^2\,, \qquad \forall\, i \tag{4.19}$$

$$b_i, p_i,\ T_i, x_{ijt}^s,\ S_{ik}^{1-2}, r_{jt}^{min},\ r_{jt}^{max}, dl_i \geq 0, x_{ijt}^s,\ S_{ik}^{1-2} \text{ binary} \tag{4.20}$$

5. HEURISTIC ALGORITHM

In the mathematical model proposed in this study, some of the constraints are non linear. Considering this and the complexity of the problem, our mathematical model can be categorized as a NP Hard problem. To have a feasible and applicable solution, we developed a heuristic algorithm which also considers the constraints of the mathematical model but does not guarantee the optimal solution. Our heuristic is composed on four main steps.

- The first step is data collection and general calculations. According to the input data, values of the variables are calculated in this first step. These calculated parameters will be input for the later steps of the heuristic.
- The second step is initial Solution for Stage 1. The success of the final result of schedule of Stage 1 depends on the quality of the initial solution. In this study, instead of finding a feasible but random initial solution we preferred to design an effective solution algorithm to find the initial solution. This initial solution will be input for the simulated annealing method later on.
- The third step is corrective algorithm – simulated annealing for stage 1. Simulated annealing approach is implemented to find the final solution and initial solution calculated in previous step is used as input to the simulated annealing search as the start point. In this step, parameters of simulated annealing algorithm, like temperature, cooling ratio..etc, are determined first and stopping criteria is given input as the number of iterations. Objective function considers the characteristics of the problem such as setup costs, due dates. Also, conditions of the machines of stage 2 are also taken into consideration in the objective function so that the risk of having infeasible or undesirable solution in stage 2 is reduced.
- The fourth step is solution for stage 2. There are a number of decision variables at the beginning of the problem. However, after we decide the solution of stage 1, some of the decision variables, such as component count, possible spool types for a job, location of the jobs, are determined. Given the solution of stage 1, an effective heuristic algorithm is developed here which considers the constraints such as due

dates, minimum location movements of the spools, best match of roller and component count. Details of these steps are described in the following sections.

5.1. Data Collection and General Calculations

To measure the efficiency of the proposed heuristics, snapshots of the live system data are gathered at two different point of time. Various scenarios are run with these two data sets and different input parameter values are used in these scenarios for test purposes. The raw data in the system is collected via a program and the data is converted into generic format to preserve the privacy of Erbakır.

During heuristic steps, calculations about the decision variables and parameters are needed several times. Instead of calculating values in every iteration, which may lead to performance problems, some of the values are calculated for all jobs and machines at the beginning of the heuristic and the calculated values are used during the iterations.

In this first step, we aim to calculate the following parameters for all job i:

- Minimum and maximum completion times in stage 1 ($Dmin_i^1$ and $Dmax_i^1$): In the second stage, jobs can be processed in parallel machines. The number of parallel machine assignment in this step is a decision variable and effects the processing time in the second stage. If the job is processed on just only one machine, the due date in the first step is the due date of the job – processing time of the job in the second step. If we divide the job to n parallel machine, the processing time will be 1/n times of the original processing time and the due date for the first step will change. In order to represent this, $Dmin_i^1$and $Dmax_i^1$ parameters are calculated. In later steps, these parameters will be used to schedule the jobs and violating these two different parameters will lead to penalty costs.
- Number of maximum parallel jobs in stage 2 ($MAXV_{ij}^2$): Spools which are used to transfer wire between stage 1 and stage 2 have different types and different capacities. Details about spool type/size limitations are pointed out in related sections previously. Number of parallel processes in stage 2 and the number of components of the job, which is decided by the machine assignment in stage 1, gives the weight

of each spool that is transferred to the second stage. In this calculation section, maximum number of parallel jobs in stage 2 with any feasible spool type is calculated.

- Feasible spool types in stage 2. (S_{ikjv}^{1-2}): The weight of a single spool between stage 1 and stage 2 is determined by the machine assignment in stage 1 and the parallel counts in stage 2. If this weigh is between acceptable ranges of a spool type, then this spool is eligible for the combination of this specific job, machine assignment in stage 1 and parallel count in stage 2.
- Number of machines according to locations in stage 2 (NM_l^2): Machines are located to different locations for both of the stages. The number of machine for each location is calculated in this step. This will be used for calculating the machine flexibility in stage 2.

Algorithm steps and flow of this section is provided in Figure 5.1.

// Set iteration count to 0:

SET v = 0.

// Visit all job/machine combinations:

LOOP AT jobs *i* and machines *j*.

// Increase iteration count

DO.

SET v = v + 1.

// Calculate Single Spool Size: Total Weight / (Component Count x Parallel Count)

$SS_{ijv}^1 = \frac{tw_i}{B_{ij}^1 \times v}$.

$PA_{ijv}^2 = 0$

Figure 5.1. Data collection and general calculations algorithm

```
// Calculations on spool types k:
LOOP AT spools k.

    // Find eligible spool types:
    IF SS^1_ijv ≥ SC^min_k and SS^1_ijv ≤ SC^max_k.
        S^(1−2)_ikjv = 1.
        PA^2_ijv = 1.
        MAXV^2_ij = v
    ELSE.
        S^(1−2)_ikjv = 0.
        PA^2_ijv = 0.
    ENDIF.

    // Calculate average duration of job i:
    DD^2_ijv = d_i − AVG_j(u^2_ij)/v

    // Calculate minimum and maximum due dates:
    IF DD^2_ijv > Dmax^1_i.
        Dmax^1_i = DD^2_ijv.
    ENDIF.
    IF DD^2_ijv < Dmin^1_i.
        Dmin^1_i = DD^2_ijv.
    ENDIF.
ENDLOOP.
IF PA^2_ijv = 0.
    EXIT for the next job i machine j
ENDIF.
ENDDO.
ENDLOOP.
```

Figure 5.1. Data collection and general calculations algorithm (continued)

// Calculate location/machine counts.
LOOP AT machines *j* stage 2.
  IF L_j^2= 1.
    ADD 1 to NM_1^2.
  ELSEIF L_j^2= 2.
    ADD 1 to NM_2^2.
  ELSEIF L_j^2= 3.
    ADD 1 to NM_3^2.
  ENDIF.
ENDLOOP.

Figure 5.1. Data collection and general calculations algorithm (continued)

5.2. Initial Solution for Stage One

The initial solution that will be input to the simulated annealing algorithm is generated with a heuristic. The heuristic described below finds a feasible initial calculation considering the priorities and constraints of the scheduling problem. Steps and flow of this section is below.

Machine flexibility index and average slack are calculated for all jobs. Average slack is the slack of a job in stage 1 according to the minimum due date. The processing time is not same for all machines of a job. In this case, we take average processing time over all the eligible machines in stage 1 into consideration.

LOOP AT jobs *i*.
  $AT_i^1 = Dmin_i^1 - AVG_j(u_{ij}^1)$.

Figure 5.2. Initial solution for stage one algorithm (section 1)

Job flexibility for each job/machine combination is calculated. Jobs in stage 1 cannot be processed on every machine due to the special properties of machines. Machine flexibility index indicates the number of eligible machines for a job. If this index is higher for a job than another, it means that this job is more flexible than the other one in stage 1 and it is preferred to assign this job to this machine if other conditions are the same.

One other important parameter for machine selection is the slack between the end of the finish time and the minimum due date. If the slack is high, for the second stage it is easier to find a free space for this job. For each job, slack depends on the processing time of the job on a machine and the first available time on the machine. Slack parameter is calculated for each job machine combination at the beginning of this step and it will be used later on.

```
LOOP AT machines j.
    SL¹ᵢⱼ = Dmin¹ᵢ − u¹ᵢⱼ − F¹ⱼ.
    IF E¹ᵢⱼ = 1.
      MF¹ᵢ = MF¹ᵢ + 1.
    ENDIF.
  ENDLOOP.
ENDLOOP.
```

LOOP AT machines j.

$SL_{ij}^1 = Dmin_i^1 - u_{ij}^1 - F_j^1$.

IF $E_{ij}^1 = 1$.

$MF_i^1 = MF_i^1 + 1$.

ENDIF.

ENDLOOP.

ENDLOOP.

Figure 5.3. Initial solution for stage one algorithm (section 2)

Jobs that have only one eligible machine are scheduled. Machine flexibility index indicates the number of eligible machines that a job can be scheduled in stage 1. If this number is one, then this job can only be scheduled on one machine only. In this step we start to schedule the jobs with only one eligible machine. If there are more than one job that has only one eligible machine, the job with smallest due date (MIN ($Dmin_i^1$)) is selected and scheduled on this single machine. During scheduling a job on a machine, some of the parameters are updated as follows: Start time is calculated as the release time of the assigned machine plus 1. End time is the release time of the selected machine summed up with 1 and processing time of the job. Release time of the machine is updated as the finish time of the scheduled job. Slack of the jobs except from the one that is scheduled should be updated as the machine is utilized by processing time of the job. Setup is important in stage one and there are sequence dependent setup costs. Setup is dependent on the radius of the jobs (i.e., closer the radius between consecutive jobs, smaller the setup cost). Average radius is calculated and updated for the selected machine. Final slack of the scheduled job according to minimum and maximum due dates in stage 1.

SORT jobs *i* and machines *j* BY MF_i^1 ASCENDING $Dmin_i^1$ ASCENDING.
LOOP AT jobs *i* and machines *j* WHERE $\mathrm{MF}_i^1 = 1$.
  ISEL = i.
  $\mathrm{ST}_i^1 = \mathrm{F}_j^1 + 1$.
  $ET_i^1 = F_j^1 + 1 + u_{ij}^1$.
  $F_j^1 = ET_i^1$.
  $SL_{iJSEL}^1 = SL_{iJSEL}^1 - u_{ISEL\ JSEL}^1$ where i <> ISEL.
  $AR_j^1 = AVG_{ji}(r_i)$.
  $SLmin_i^1 = Dmin_i^1 - ET_i^1$.
  $SLmax_i^1 = Dmax_i^1 - ET_i^1$.
  $SC_i^1 = (r_i - r_{i-1})$.
ENDLOOP.

Figure 5.4. Initial solution for stage one algorithm (section 3)

Iteration index is set to 0. *c* is the iteration index which indicates the number of eligible machines and *Cmax* is the number that indicates the maximum available machine for any single job. To schedule jobs, each job group with different number of eligible machine is visited starting with the group with minimum eligible machine. Jobs are sorted by machine flexibility index and average tardiness ascending.

SET *c* = 0
DO *Cmax* times.
  SET *c* = *c* + 1.
  SORT jobs *i* and machines *j* BY MF_i^1ASC AT_i^1 ASC.

Figure 5.5. Initial solution for stage one algorithm (section 4)

Jobs that have more than one eligible machine are scheduled. Job group with eligible machine number *c* is selected. Within this group, the job having least slack value is scheduled on one of the alternative machines of the job. Machine selection for scheduling this job depends on two factors:

- Radius difference between individual job and machine average (i.e., setup cost).
- Slack of the job on a machine.

These two parameters are weighted and machine selection index (MS_{ij}^1) is calculated and the machine with the smallest index value is selected. After determining the job and the machine, selected job is scheduled on the selected machine. Parameters are updated after scheduling the job on the machine as stated on the previous step. When all the jobs of group with eligible machine number *c* are scheduled, next group selected. At the end, all the jobs are scheduled and initial solution is obtained.

Table 5.1. Weight parameters of initial heuristic in stage 1

W1	5000
W2	0,001

LOOP AT jobs i WHERE $MF_i^1 = c$.

  isel = *i*.

  LOOP AT machines j

    $MS_{ij}^1 = \left(W1 \times ABS\left(r_i - AR_j^1\right)\right) + (W2 \times SL_{ij}^1)$.

    IF MSMIN $< MS_{ij}^1$.

      MSMIN = MS_{ij}^1.

      jsel = *j*.

    ENDIF.

  ENDLOOP.

  $\mathrm{ST}_i^1 = \mathrm{F}_j^1 + 1$.

Figure 5.6. Initial solution for stage one algorithm (section 5)

$ET_i^1 = F_j^1 + 1 + u_{ij}^1$.

$F_j^1 = ET_i^1$.

$SL_{iJSEL}^1 = \mathrm{SL}_{iJSEL}^1 - u_{ISEL\ JSEL}^1$ WHERE $i \neq isel$.

$AR_j^1 = AVG_{ji}(r_i)$.

$SLmin_i^1 = Dmin_i^1 - ET_i^1$.

$SLmax_i^1 = Dmax_i^1 - ET_i^1$.

ENDLOOP.

ENDDO.

Figure 5.6. Initial solution for stage one algorithm (section 5) (continued)

5.3. Corrective Algorithm

After finding the initial solution for the scheduling problem in stage 1, at this step initial solution will be improved and eventually we will reach the final solution for stage 1. Simulated annealing technique, a meta-heuristic, will be employed in this study to determine a scheduling policy so as to minimize the objective function. The results of the first stage will be used in the second stage as input but during scheduling in this step, values of the variables in the second step will also be check in order to sustain the feasibility.

Simulated annealing was introduced first by Kirkpatrick et al. [40] as a method to solve combinatorial optimization problems. In general, search algorithm in difficult combinatorial problems first generates neighborhood solutions from a current solution, and then, continues to move to one of them until defined conditions are fulfilled. In simulated annealing, moves are not only from a worse solution to a better solution but also it probabilistically allows deteriorating the solution by movements so that a neighborhood solution which is not in the local optimum can be tried. This prevents the iterations of the algorithm to get stuck in a local optimum solution.

To start the simulated annealing heuristic we need an initial solution (I), which we calculated in the first step. Initial temperature (T), iteration count to change the temperature (IT), iteration count to stop the algorithm (IS) and cooling ratio (α) are the other parameters of simulated annealing approach. Temperature controls the acceptance of a worse solution. At early stages of iterations, temperature is high and acceptance probability of a worse solution is also high. At later iterations, since the temperature goes lower values, the flexibility of accepting worse solution is also diminishes. Iteration number shows the number of repetitions until the solution reaches to a stable state and reaching the maximum iteration number is the termination criteria of the algorithm.

Based on the values of temperature a new neighborhood solution is generated in the heuristic. This neighborhood solution is the new solution if the value of the objective function is better than the value of the objective function previously. However, even though the new objective function value is worse than the previous one, it may also be accepted with a probability based on $e^{-\Delta/T}$ where Δ is the difference between current solution and the best solution so far and T is the temperature. According to this formula, higher the temperature (i.e., first iterations), higher the probability of accepting the worse solution. Likewise, lower the Δ (i.e., the difference between the current and best objective value), higher the probability of accepting the worse solution. This leaves the possibility of finding a global optimal solution apart from a local optimum. After a number of iterations, the temperature diminishes and the probability of accepting worse solution decreases.

Simulated annealing algorithm for a minimization problem is represented in Figure 5.7. Figure 5.8. also gives the process flow of the algorithm.

```
START with initial solution I.
    DEFINE (T ; IT ; IS ; α ).
    OBJ(X) = OBJ(I).
    DO.
        FOR Y = 1 to IT DO.
            Find a neighborhood solution.
            Calculate OBJ(N) = Objective function value of new neighborhood
solution.
```

Figure 5.7. Simulated annealing algorithm

Δ = OBJ(N) – OBJ(X).

IF (Δ > 0) OR ($e^{-\Delta/T}$ > RND (0, 1))

X = Y {➔ accept the movement}

ENDIF.

ENDFOR.

UPDATE (T) {➔ T = α x T}.

UNTIL(IS).

END.

Figure 5.7. Simulated annealing algorithm (continued)

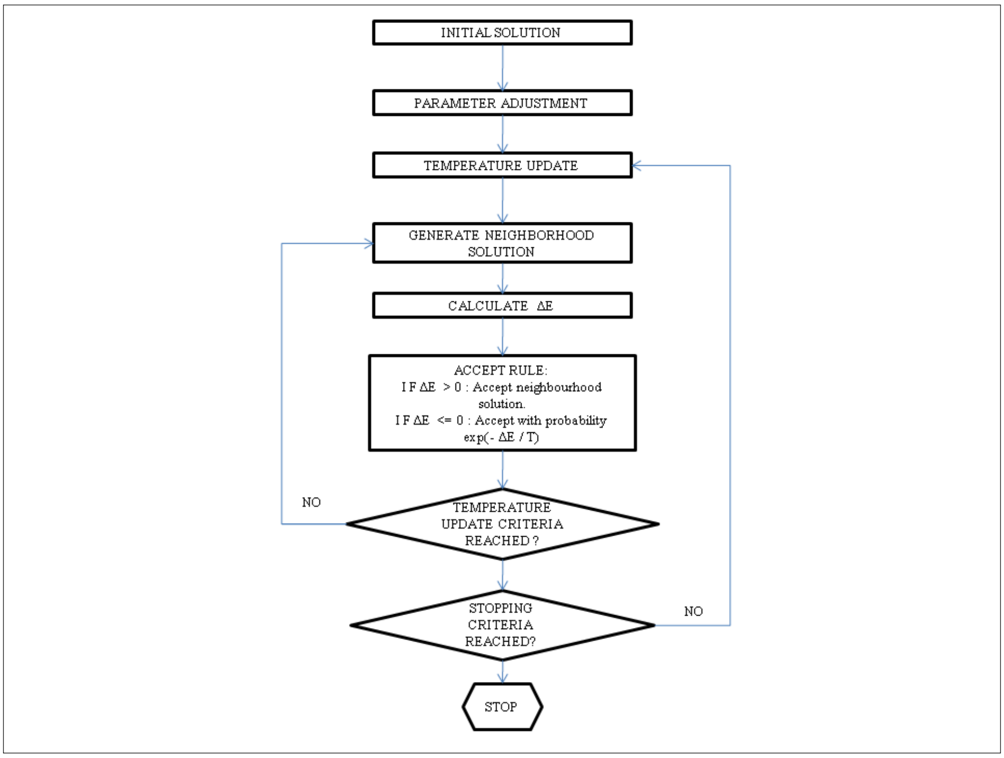

Figure 5.8. Simulated annealing flow

There are several factors to be decided first in order to apply SA to practical problems. First of all, it is necessary to define a procedure to generate neighborhood solutions from a current solution. In order to efficiently generate these solutions, parameters such as initial temperature (T), iteration count to change the temperature (IT), iteration count to stop the algorithm (IS) and cooling ratio (α) should be appropriately decided. The combination of these parameters needs to be adjusted based on problems to achieve good solutions [37].

5.3.1. Determination of Parameters

Simulating annealing is a meta-heuristic method and the result of this heuristic is based on the parameters both at the beginning of the algorithm and during the iterations. Therefore, parameter tuning is an important issue in order to get an effective solution. Main parameters of the simulated annealing are:

- Initial temperature (T),
- Cooling ratio (α),
- Iteration number to change the temperature (IT),
- Iteration number to stop the algorithm (IS).

Several tests are performed with a data set in order to identify the parameters. Each time, we changed the value of one of the parameters and kept the values of other parameters fixed. Each parameter is tested with several different values (between 5 to 12 times each for each parameter).

5.3.1.1. Initial Temperature (T). Initial temperature parameter affects the acceptance ratio of the worse solution at the beginning of the algorithm. If the temperature is high, the probability of accepting a worse solution becomes also high, which leads to have longer iteration times but also avoids the risk of trapping to a local optimal solution. To find the best parameter for temperature, 11 test runs are performed with data set 1. After the first run, the average value of delta is calculated. Temperature values of the rest of the test cases are calculated so as to obtain stated initial probability to accept a worse solution value. Values of the parameters of all the test cases and the graphical display of the selected test run are in following table. The results of all test runs are presented in Appendix A and the best value for the temperature parameter is set to 110.000 after the tests.

Table 5.2. Temperature parameter tests – all tests

Test #	Temp. (T)	Cooling ratio (α)	Iter # change temp. (IT)	Iter # to stop (IS)	Initial probability to accept worse solution
1	5.000	0,95	625	3.500	<1%
2	25.000	0,95	625	3.500	2%
3	110.000	0,95	625	3.500	40%
4	125.000	0,95	625	3.500	45%
5	145.000	0,95	625	3.500	50%
6	200.000	0,95	625	3.500	60%
7	230.000	0,95	625	3.500	65%
8	280.000	0,95	625	3.500	70%
9	345.000	0,95	625	3.500	75%
10	450.000	0,95	625	3.500	80%
11	615.000	0,95	625	3.500	85%

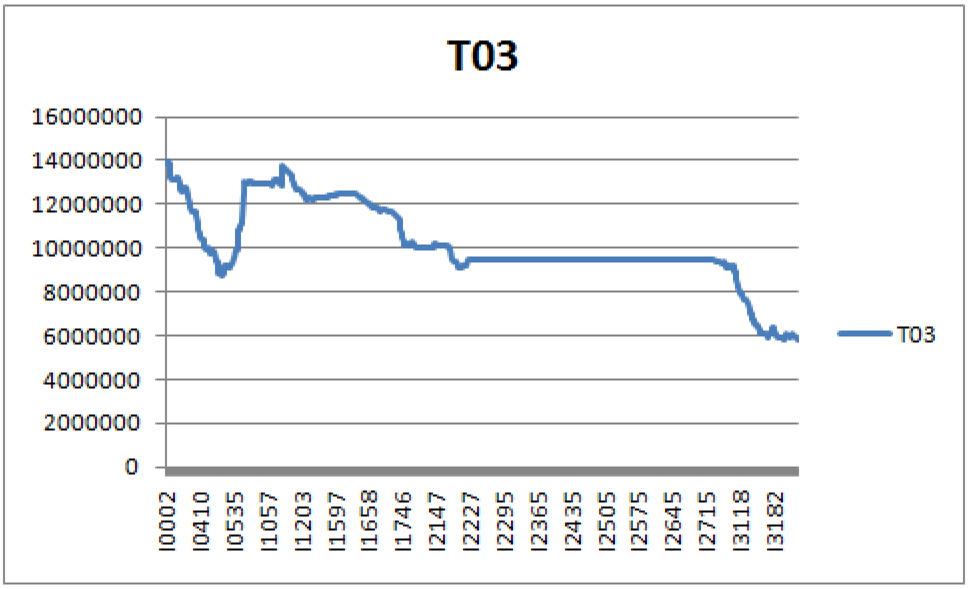

Figure 5.9. Temperature parameter tests – selected test

<u>5.3.1.2. Cooling Ratio (α).</u> Cooling ratio parameters effect the temperature value identifies how rapidly the temperature decreases after certain number of iterations. If we keep this parameter high, the temperature will not decrease quickly and the probability of accepting a worse solution will not diminish rapidly as the iteration number increases. Controversially, if the cooling ratio is low, the temperature will converge to zero quickly

and the acceptance ratio will also converge to zero. After deciding the initial temperature, cooling ratio factor is tested with 12 test cases and the best cooling ratio is selected. Values of the parameters of all the test cases and the graphical display of the selected test run are as follows:

Table 5.3. Cooling ratio parameter tests – all tests

Test #	Temp. (T)	Cooling ratio (α)	Iter # change temp. (IT)	Iter # to stop (IS)
12	110.000	0,99	625	3.500
13	110.000	0,96	625	3.500
14	110.000	0,93	625	3.500
15	110.000	0,90	625	3.500
16	110.000	0,87	625	3.500
17	110.000	0,85	625	3.500
18	110.000	0,80	625	3.500
19	110.000	0,75	625	3.500
20	110.000	0,70	625	3.500
21	110.000	0,60	625	3.500
22	110.000	0,50	625	3.500
23	110.000	0,40	625	3.500

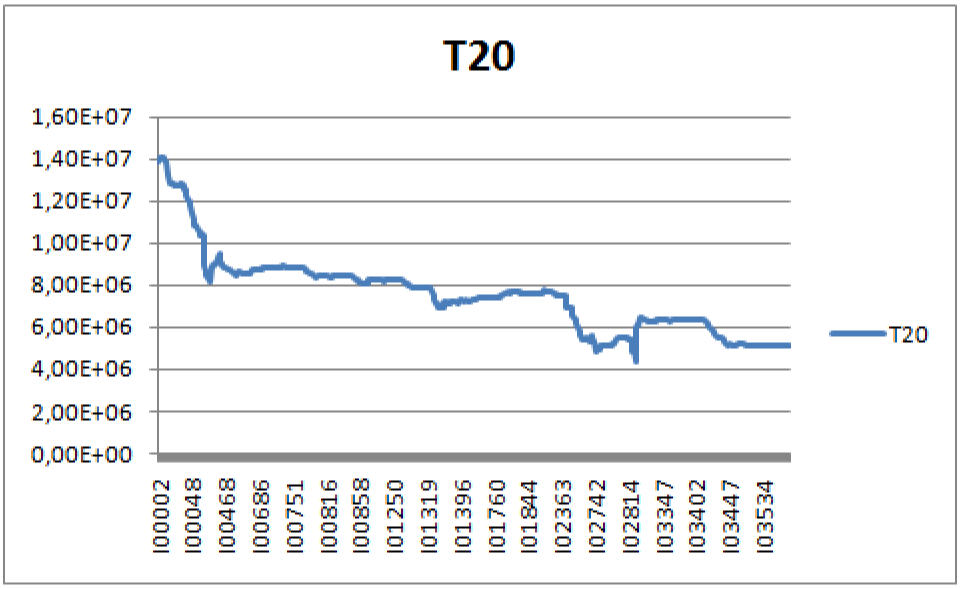

Figure 5.10. Cooling ratio parameter tests – selected test

The results of the iterations are presented in Appendix A and the best value for the cooling ratio parameter is set to 0,70 after the tests.

5.3.1.3. Iteration Number to Change the Temperature (IT). After a certain amount of iterations, the temperature should be decreased .The starting point of this parameter is the number of average jobs in stage 1. Starting from this point, 4 other values are tested and the best value is selected. Values of the parameters of all the test cases and the graphical display of the selected test run are as follows:

Table 5.4. Iteration # to change the temperature tests – all tests

Test #	Temp. (T)	Cooling ratio (α)	Iter # change temp. (IT)	Iter # to stop (IS)
20	110.000	0,70	625	3.500
24	110.000	0,70	550	3.500
25	110.000	0,70	475	3.500
26	110.000	0,70	400	3.500
27	110.000	0,70	325	3.500

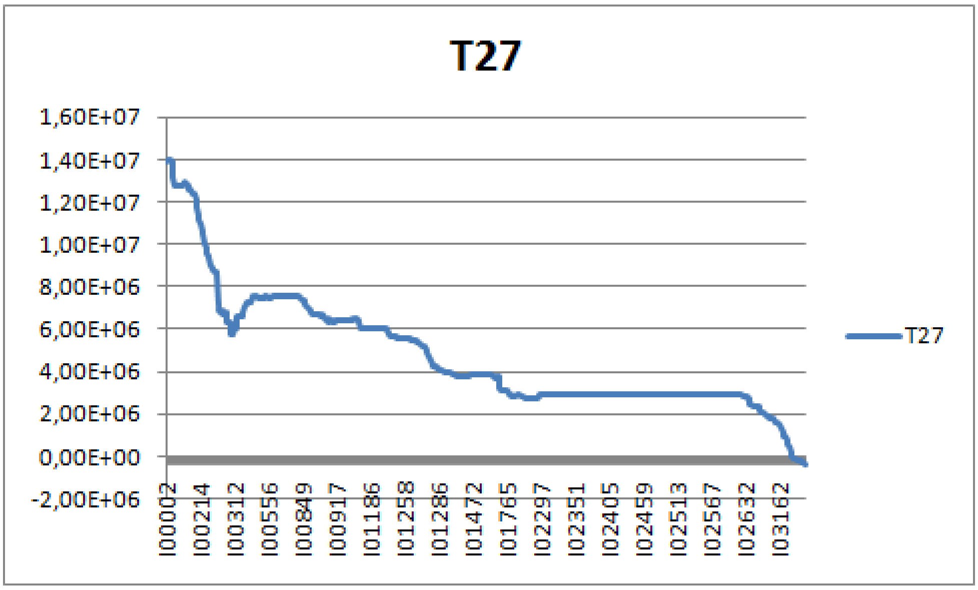

Figure 5.11. Iteration # to change the temperature tests – selected test

The results of the iterations are presented in Appendix A and the best value for the Iteration # to change the temperature is set to 325 after the tests.

5.3.1.4. Iteration Number to Stop the Algorithm (IS). Simulated annealing is basically a search algorithm and it searches neighborhood solutions within a logic and accepts the solution or not. The result of the algorithm depends on the run time and therefore on the iteration number. The more iteration we perform the more probability that we will reach a better solution. However, the computation time is also a critical factor for real-life problems. In many cases after a certain amount of time, obtaining a unit improvement in objective function requires considerably high times. Iteration number to stop the algorithm decides the appropriate number of total iterations for the simulated annealing algorithm. 6 different values are tested for this parameter. Values of the parameters of all the test cases and the graphical display of the selected test run are as follows:

Table 5.5. Iteration # to stop the algorithm – all tests

Test #	Temp. (T)	Cooling ratio (α)	Iter # change temp. (IT)	Iter # to stop (IS)	Duration (min.)
27	110.000	0,70	325	3.500	12
28	110.000	0,70	325	5.000	17
29	110.000	0,70	325	15.000	54
30	110.000	0,70	325	17.500	60
31	110.000	0,70	325	20.000	67
32	110.000	0,70	325	30.000	98

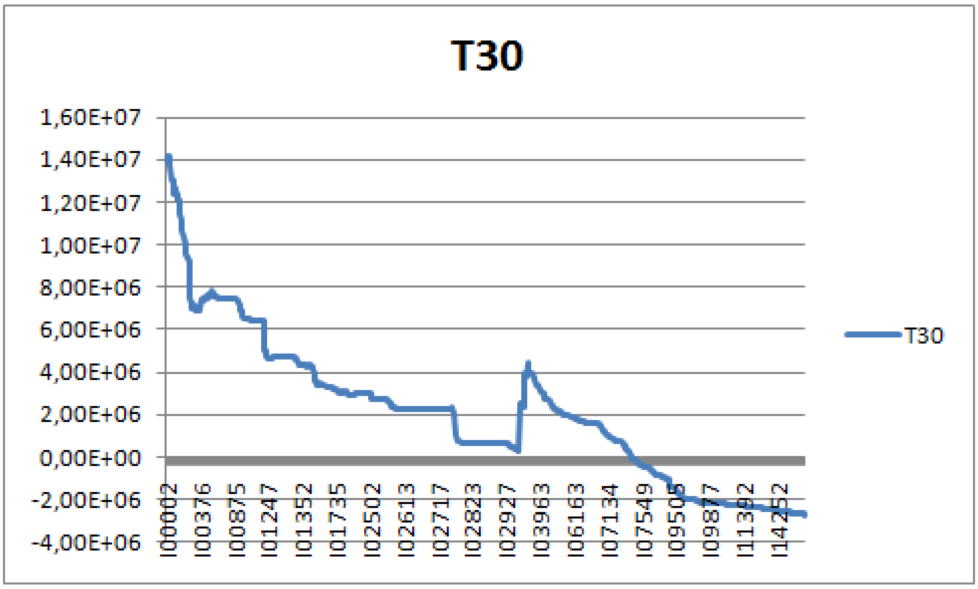

Figure 5.12. Iteration # to stop the algorithm - selected test

The results of the iterations are presented in Appendix A. The quality of the test 30 is high enough and the computation time is fair. Although the result of the test run 32 is better, test run 30 is preferred in this study.

As a result, algorithm is run with the following parameters:

Table 5.6. Simulated annealing final parameters

Temp. (T)	Cooling Ratio (alpha)	Iter # change temp. (IT)	Iter # to stop (IS)
110.000	0,70	325	17.500

5.3.2. Defining Objective Function

Our problem is composed of 2 stages. In the solution procedure recommended in this study, stages are solved separately and the solution of the first stage becomes input to the second stage and second stage is solved according to the first one. During scheduling in first step, we consider cost categories about the special properties of the problem like setup times, due dates. In this case, there is a risk of having a solution input for stage 2 which leads to infeasible or inappropriate solutions in stage 2. Therefore, we have added parameters about the solution quality in stage 2 to the objective function of stage 1 and during objective function calculation these parameters are also taken into account. Here are the objective function segments and stage 1 related parameters.

Setup cost is calculated as the difference of radiuses of adjacent jobs on one machine. The sum of all the radius differenced multiplied by setup cost weight gives the weighted value of setup cost overall.

$$+ W3 \times \sum_{j} \sum_{i} (r_i - r_{i-1}) \tag{5.1}$$

The more slack to the due dates we have in stage 1, more flexibility we will obtain in stage 2. Since this is a minimization function, we reduce the weighted total slack of the solution from the objective value.

$$- W4 \times \sum_{i} (Dmin_i - ET_i^1) \tag{5.2}$$

Slack to the maximum due parameter is similar to the previous one but the weight of this parameter is higher than the slack to the minimum due date. If a job is scheduled later than the maximum due date in stage 1, it is impossible to schedule the job in the second stage without having a delay to the customer due date of the job. Therefore we should give more penalty of not obeying the maximum due date during scheduling.

$$- W5 \times \sum_{i} (Dmax_i - ET_i^1) \tag{5.3}$$

Solution with minimum location movements between two stages is the critical objective of the algorithm and it is the first stage 2 related parameter. If we do not consider this situation at this stage, there is a risk of having overload on a single location in stage 2. To avoid this, average capacity loads on a unit machine for each location is calculated and deviation of the unit machine load in stage 2 is tried to be minimized.

$$LOCAVG = \left(\frac{AVG_{1j}(u_{ij}^2)}{\mathrm{NM}_1^2} + \frac{AVG_{2j}(u_{ij}^2)}{NM_2^2} + \frac{AVG_{3j}(u_{ij}^2)}{NM_3^2} \right) /3 \tag{5.4}$$

LOCAVG parameter shows the average unit machine load over three physical location.

$$\begin{aligned} LOCCOST = \;& ABS\left(\frac{AVG_{1j}(u_{ij}^2)}{NM_1^2} - LOCAVG \right) \\ &+ ABS\left(\frac{AVG_{2j}(u_{ij}^2)}{NM_2^2} - LOCAVG \right) \\ &+ ABS\left(\frac{AVG_{3j}(u_{ij}^2)}{NM_3^2} - LOCAVG \right) \end{aligned} \tag{5.5}$$

LOCCOST parameter is the sum of absolute deviation between average unit machine load over three physical locations and machine load of each location. Small deviation number represents that unit machine load of locations in stage 2 are close to each other according to the scheduling solution in stage 1. By this way, we increase the chance of assigning the job to a machine in stage 2 having the same location of the machine assignment in stage 1.

$$+ W6 \times \mathrm{LOCCOST} \tag{5.6}$$

Spool – component count compatibility parameter is the last parameter. Spool and component counts of jobs in stage 1 and stage 2 should be as close as possible in order to maintain cost reduction. Machine flexibility of stage 2 (MF_i^2) shows the number of machines in stage 2 having same or only one different spool-component count machines according to the machine assignment in stage 1. If the difference is more than 1, this machine is not included in machine flexibility count for this job. The objective is to assign the jobs in stage 1 to a machine which generates maximum machine flexibility in stage 2.

$$+ W7 \times \mathrm{MF}_i^2 \tag{5.7}$$

Eventually, the objective is to minimize the following function with simulated annealing method :

$$\begin{aligned} &+ W3 \times \sum_j \sum_i (r_i - r_{i-1}) - W4 \times \sum_i (Dmin_i - ET_i^1) \\ &- W5 \times \sum_i (Dmax_i - ET_i^1) + W6 \times \mathrm{LOCCOST} + W7 \times \mathrm{MF}_i^2 \end{aligned} \tag{5.8}$$

Weights of the parameters are decided with the planner of the Erbakir. Planner, according to his business priorities, identifies the weight of each parameter and the heuristic considers these parameters. Table 5.7. shows these weights.

Table 5.7. Weight parameters of simulated annealing objective function

W3	6.000,0000

Table 5.7. Weight parameters of simulated annealing objective function (continued)

W4	0,0095
W5	0,0125
W6	8,0000
W7	0,5000

5.3.3. Generation of Neighborhood Solution

There are several techniques of choosing neighborhood selection in literature. Choosing better neighborhood solution for each iteration, affects the quality of the general solution [37].

In this study, instead of choosing the neighborhood solution randomly, we developed a logic behind the neighborhood selection. According to our logic, which is described below, at each iteration most suitable two jobs are identified and the first job is inserted behind the second job. After the insert move, parameters setup cost, start time, end time, slack, location selection of the effected jobs are updated and new objective function value is calculated.

DO (IS) TIMES.

// Jobs are sorted by maximum slack and setup cost parameters.

SORT jobs BY $SLmax_i^1$ SC_i^1 DESCENDING.

// Select the job with index (IS).

READ jobs *i* with index (IS).

// Select the job with least slack regarding maximum due date

Figure 5.13. Neighborhood solution algorithm (section 1)

If the slack according to the maximum due date is lower than zero, which means this job is late even if we split the job into maximum number of parallel machines in stage 2, this job is selected as job 1.

```
IF $SLmax_i^1 \leq 0$.
        ASSIGN job1.
    // Select the job with least slack regarding minimum due date:
```

Figure 5.14. Neighborhood solution algorithm (section 2)

If the slack according to the maximum due date is not lower than zero but slack according to the maximum due date is lower than zero, this job is selected as job 1.

```
ELSEIF $SLmin_i^1$ <= 0.
    SORT jobs *i* BY $SLmin_i^1$ $SC_i^1$ DESCENDING.
    READ jobs *i* with index IS.
ASSIGN job1.
// Select the job with highest setup cost
```

Figure 5.15. Neighborhood solution algorithm (section 3)

If the job does not fulfill the above conditions, jobs are sorted only by setup costs and the one with index (IS) is selected. As job1

```
ELSE.
    SORT jobs *i* BY $SC_i^1$ DESCENDING.
    READ jobs *i* with index IS. ASSIGN job1.
ENDIF.
// Calculate the setup differences between the selected job1 and other jobs.
LOOP AT jobs *i*.
    $SC_i^1 = ABS(r_i - r_{job1})$.
ENDLOOP.
// Find a place for job 1
```

Figure 5.16. Neighborhood solution algorithm (section 4)

Find the best place to insert the job 1 considering the setup times. Among the equal setup cost places, the one with maximum slack time selected as job 2.

```
SORT jobs i BY SC_i^1 ASCENDING SLmin_i^1 DESCENDING.
    READ jobs i with index 1.
    ASSIGN job2.
ENDDO.
```

Figure 5.17. Neighborhood solution algorithm (section 5)

After selecting jobs, the first job is inserted just before the second job and parameters are updated. The objective value of this temporary solution is evaluated and compared with the best solution so far. If this solution is better, this neighborhood solution is accepted and objective function value is updated. If this solution is worse, depending on the difference of the current and existing solution and temperature, it is possible to accept the worse solution stochastically in order to avoid to be trapped in local optimal solution.

Until the stopping criteria, this simulated annealing algorithm runs and at the end final solution is obtained. This final solution is the input of stage 2. The details of the result of stage 1 compared with planners solution can be found in experiments section.

5.4. Solution for Stage Two

Solution procedure in stage 2 is composed of two levels. At first level, given the solution in stage 1, some of the parameters in stage 2 will be updated. At the second level, a solution heuristic for the problem in stage 2 will be run and the final solution will be the output of this stage.

Jobs in second stage cannot start before the end time of the first state. Therefore, first of all, release time of the jobs in second stage are calculated and updated. Some of the jobs do not have the first stage and production starts from the second stage due to either product structure or the production in the first stage have already been finished at the time of data collection. The release time of those jobs are considered as 0. At this point, average slack of the jobs in stage 2 considering the release time, average processing time and due date of the jobs are also calculated.

```
LOOP AT jobs i.
   IF MF_i^1 = 0.
         RT_i^2 = 0.
      ELSE.
      RT_i^2 = ET_i^1
   ENDIF.
   T_i = d_i − RT_i^2 + AVG_j(u_ij^2).
ENDLOOP.
```

Figure 5.18. Solution for stage two algorithm (section 1)

Jobs are sorted by their release time ascending. In the first step, the objective function of the simulated annealing algorithm contains factors about the due dates of the jobs. Therefore, the end time of the jobs having earlier due dates are most probably located at the earlier positions on the machines and the release dates of those jobs are earlier than the others. Also, some of the jobs in stage 2 do not have the first stage of production and their release times are set to zero. At this point of the algorithm, we start scheduling jobs having earlier release dates in order to fill the earlier timelines of the machines at stage 2.

```
SORT jobs i BY RT_i^2 ASCENDING.
```

Figure 5.19. Solution for stage two algorithm (section 2)

Jobs are scheduled sequentially after sorting by the release time. The one with earliest release time is selected first. After job selection, a machine is chosen among the alternatives. Machine selection considers penalties that will occur during transfer of spools between stages and due to not obeying the due date. Considering all these factors, a machine penalty index is assigned for the machines for this scheduled job and the machine having least penalty is chosen. In this way, unnecessary movements between locations, spool and component count compatibility and due dates are considered and the best machine is selected.

```
LOOP AT jobs i.
LOOP AT machines j for job i.

// Calculate location change penalty
    CALCULATE LP^2_ij.

// Calculate spooler – component count compatibility penalty
    CALCULATE SP^2_ij.
    UPDATE T_i
    PM^2_ij = W8 × LP^2_ij + W9 × SP^2_ij + W10 × T_i
ENDLOOP.
CHOOSE machine j with minimum PM^2_ij.
```

Figure 5.20. Solution for stage two algorithm (section 3)

Jobs in stage 2 can be scheduled in parallel machines when necessary in order to obey the close due dates. This algorithm tries to schedule the jobs in stage 2 as one job first. If the slack of the job is negative, which means the job is tardy; this job is divided into parallel machines considering the spool type eligibility. Initial slack of the jobs are calculated in the previous step and this is the start point of the loop.

```
IF T_i < 0.
```

Figure 5.21. Solution for stage two algorithm (section 4)

Maximum possible parallel count for this job has already been identified at the beginning of stage 1 data collection and general calculations section. Since this job is tardy when we schedule it as a single job to only one machine, we should try to make the production in parallel machines and divide the job into several items. However, in this case, the maximum parallel count and machine assignment combination may not allow selecting among all the alternative machines in stage 2 because of the spool type constraints. In order to overcome this situation, machines in stage 2 are evaluated and revealed whether there is at least one eligible machine when we consider the component count in stage 1, maximum parallel split and weight of the job. If there is no machine that fulfils this requirement, then the maximum count is reduced by 1 and machines in stage 2 are again searched for eligibility.

```
DO MAXV_ij^2 times where i and j are determined.
    v = MAXV_ik^2.
    LOOP AT machines j for job i.
        LOOP AT spools k WHERE S_ikjv^(1-2) = 1.
        ENDLOOP.
        IF FOUND.
            EXIT.
        ENDIF.
    ENDLOOP.
    IF FOUND.
        EXIT.
    ELSE.
        MAXV_ik^2 = MAXV_ik^2 - 1.
    ENDIF.
  ENDDO.
```

Figure 5.22. Solution for stage two algorithm (section 5)

After the item count is identified in previous section, each item is scheduled separately to the alternative machines. The processing time of each item is calculated by dividing the processing time of the job to the parallel count. Eventually, jobs are scheduled and parameters like start time, end time, machine load..etc are updated.

```
u_ijv^2 = u_ij^2 / MAXV_ki^2
        IF RT_i^2 < F_j^2.
            ST_i^2 = RT_i^2
        ELSE.
```

Figure 5.23. Solution for stage two algorithm (section 6)

$ST_i^2 = F_j^2$

ENDIF.

$ET_i^2 = ST_i^2 + u_{ijv}^2$.

$F_j^2 = ET_i^2$

$T_i = \mathrm{T}_i - u_{ijv}^2$

ENDIF

Figure 5.23. Solution for stage two algorithm (section 6) (continued)

The overall objective function includes parameters about stage 1 and stage 2. The final solution of the stage 2 and the solution of stage 1 which is also input for stage 2 are evaluated together to get the final cost of the solution. The elements of the function are:

- Setup costs in stage 1 ($\mathrm{SC_i^1}$). Setup costs are calculated as the difference of radiuses of the adjacent jobs in a same machine at stage 1.
- Location changes between stages ($\mathrm{LP_i^{1-2}}$). If a job is produced in a specific location in stage 1 and the production in stage 2 is not in that location, then a penalty for this job applies. This penalty is the location transfer penalty and total number of location transfers shows this cost.
- Spooler / component count compatibility. ($\mathrm{SP_i^2}$). Component count decision in stage 1 and spooler count decision in stage 2 are depends on the machine assignments. If the component count is smaller than the spool count, some of the spools of the machine in stage 2 will not be utilized and the penalty of this situation depends on the difference of the counts. When the spool count is smaller than the component count, the penalty is twice the difference since in this case additional spoolers should be carried to the location.
- Tardiness (T_i). If the end time of the job in stage 2 is greater than the due date, this job is considered as tardy and it is important to minimize the tardiness.

The objective function is the weighted sum of the above parameters and can be formulated as follows:

$$\sum_{\mathrm{i}} \mathrm{W3} \times SC_i^1 + \mathrm{W8} \times \mathrm{LP}_{\mathrm{i}}^{1-2} + \mathrm{W9} \times SP_i^2 + \mathrm{W10} \times T_i^2 \tag{5.9}$$

6. EXPERIMENTATION

The proposed algorithms in this thesis have been programmed in the customers system. Erbakır uses SAP ECC system and ERP system in combination with SAP SCM system for the planning purposes. Heuristics are programmed in SAP SCM system using SAP's internal programming language ABAP. Programs are run in live system and the data used in these algorithms are collected from the same system also.

Coded program composed of more than 2500 lines. 18 new database tables and their relations are created either to collect data from the system or to write the result of the algorithm from stage 1 and stage 2. 158 variables are created within the heuristic program. Screenshot of the parameter selection screen of the written program is below.

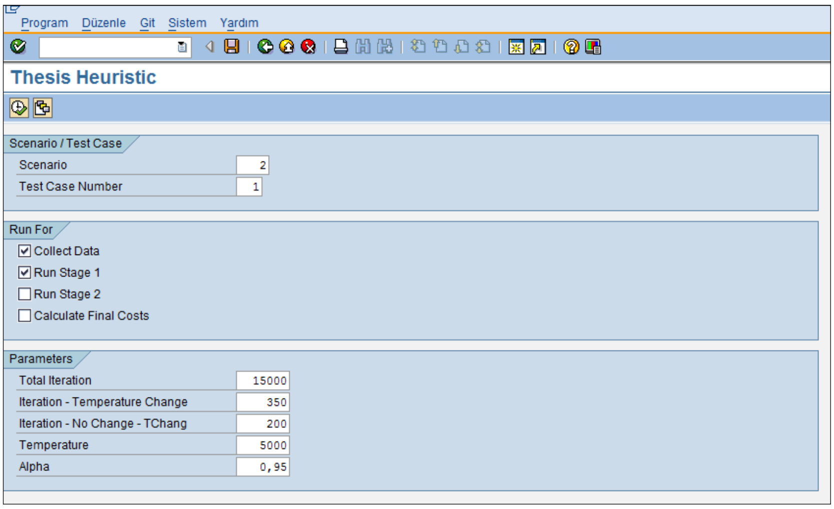

Figure 6.1. Parameter screen of the program

Three real-life data scenarios are downloaded to custom database tables and there is one week of time interval between the data sets in order to have different sets of data. Two of these sets also have the current scheduling data on the SCM system. This current data is the output of SAP SCM system and also influenced by the planners manual adjustments. Some basic information about these data sets is as follows:

Table 6.1. Problem size information

Data Sets	# of jobs			# of machines		Max. # of alternative for a single job		Min. # of alternative for a single job		**# of spool types**
	Stg1	**Stg2**	**Total**	**Stg1**	**Stg2**	**Stg1**	**Stg2**	**Stg1**	**Stg2**	
Set 1	665	787	820	20	153	15	148	1	1	5
Set 2	643	743	768	21	153	16	148	1	1	5
Set 3	673	747	809	21	153	16	148	1	1	5

Set 1 is used for test purposes of the parameters of the algorithm. Set 1 has 665 jobs in stage 1 and 787 jobs in stage 2. Number of alternative machines in stage 1 for set 1 is 20. On the other hand number of alternative machines in stage 2 for set 1 is 153 which is considerably high for a scheduling problem. 5 different spools can be used in the problem set and these spool types are alternative to each other during transfer of spools between stage one and stage two. As explained in the parameter tuning section, calculating initial solution, simulated annealing approach for the first step and scheduling heuristic in the second stage uses several different parameters. Selected parameters are summarized in Table 5.6.

After identifying parameters, Set 2 and Set 3 are run for all the steps of the algorithm and final solution is calculated. These sets of data also have the current scheduling information that is planned in the system and adjusted by the planner. The cost of the current plan using the same logic and parameters as the heuristic approach uses is also calculated and compared.

The first solution step of the algorithm is obtaining initial solution for the first stage. In this step we concentrate on the important parameters and aimed to get a better start point to the simulated annealing algorithm. The parameters that we used in this step are explained in related sections. Getting the solution as input, simulated annealing algorithm adjusts and improves the solution. Following table and graphic shows the change of setup costs and slacks to the minimum and maximum due dates between initial solution and simulated annealing algorithm based on machine level.

Table 6.2. Initial – corrected solution comparison (Stage 1 – Set 2)

INITIAL SOLUTION				CORRECTED SOLUTION			
Mac.	Setup Total	MinDue Slack	MaxDue Slack	Mac.	Setup Total	MinDue Slack	MaxDue Slack
M01	0,108	17.571.828	18.571.671	M01	0,104	46.091.585	47.602.475
M02	0,220	27.755.619	29.934.649	M02	0,130	47.584.428	49.089.995
M03	0,100	53.782.924	59.065.036	M03	0,173	73.343.945	79.221.899
M04	0,080	24.420.011	26.689.987	M04	0,148	23.239.678	27.152.436
M05	0,080	25.477.514	27.457.335	M05	0,097	58.521.574	64.965.310
M06	0,080	93.170.937	93.670.065	M06	0,220	94.083.486	94.622.212
M07	0,340	35.416.555	38.139.560	M07	0,457	85.256.586	92.899.570
M08	0,740	6.380.942	6.483.316	M08	0,659	75.921.124	77.995.641
M09	0,080	2.379.972	2.478.700	M09	0,080	3.930.228	4.449.617
M10	0,030	178.598	409.675	M10	0,000	0	0
M11	0,084	13.422.559	15.472.683	M11	0,057	17.159.987	20.775.566
M13	0,250	14.910.715	15.298.063	M13	0,168	130.867.511	134.799.554
M14	0,160	1.458.647	1.479.561	M14	0,160	1.458.647	1.479.561
M15	0,030	7.624.021	7.996.323	M15	0,030	3.366.667	3.625.524
M16	0,103	45.414.268	50.694.034	M16	0,115	17.605.973	19.597.468
M17	0,266	76.874.131	88.473.493	M17	0,114	41.523.453	45.864.312
M18	0,466	159.254.314	166.919.952	M18	0,363	117.098.557	120.616.198
M19	0,330	51.340.806	52.403.719	M19	0,040	5.469.362	5.694.040
M20	0,356	167.715.499	172.333.841	M20	0,096	148.756.694	152.930.938
M21	0,243	225.277.380	229.708.465	M21	0,214	108.029.344	109.779.401
Avg.	**0,207**	**52.491.362**	**55.184.006**	**Avg.**	**0,171**	**54.965.441**	**57.658.085**

Table 6.3. Initial – corrected solution comparison (Stage 1 – Set 3)

INITIAL SOLUTION				CORRECTED SOLUTION			
Mac.	Setup Total	MinDue Slack	MaxDue Slack	Mac.	Setup Total	MinDue Slack	MaxDue Slack
M01	0,116	16.527.466	17.330.195	M01	0,176	27.722.935	29.422.517
M02	0,306	23.009.430	24.978.512	M02	0,120	124.688.194	125.690.578

Table 6.3. Initial – corrected solution comparison (Stage 1 – Set 3) (continued)

M03	0,232	44.823.559	50.904.314	M03	0,234	43.552.280	47.873.876
M04	0,124	30.954.561	33.706.338	M04	0,110	26.694.596	31.832.384
M05	0,156	39.608.145	42.293.578	M05	0,112	53.994.993	58.633.474
M06	0,085	-1.887.986	-1.603.696	M06	0,070	-1.033.353	-749.063
M07	0,264	31.802.575	33.023.499	M07	0,224	83.466.663	87.609.163
M08	1,232	310.892.172	310.892.172	M08	0,409	199.302.214	200.281.708
M09	0,134	3.675.668	3.829.390	M09	0,080	3.147.835	3.384.818
M10	0,057	2.884.661	3.725.681	M10	0,030	1.941.909	2.314.870
M11	0,033	19.772.025	26.902.783	M11	0,087	14.116.845	21.632.401
M12	0,175	37.300.633	38.642.099	M12	0,212	54.077.640	55.993.292
M13	0,132	10.429.093	11.706.123	M13	0,040	12.166.730	14.234.714
M14	0,060	8.451.245	8.609.645	M14	0,040	8.655.417	8.813.817
M15	0,210	7.399.490	7.844.484	M15	0,030	4.204.500	4.267.345
M16	0,030	-354.568	-354.568	M16	0,000	0	0
M17	0,818	153.159.435	159.874.566	M17	0,496	153.012.007	158.992.268
M18	0,152	13.567.725	15.706.416	M18	0,070	26.838.580	29.829.687
M19	1,166	111.053.250	112.853.113	M19	0,898	111.361.329	113.121.440
M20	0,370	129.966.387	142.569.430	M20	0,236	68.081.191	73.598.582
M21	0,136	223.693.053	228.533.151	M21	0,151	182.242.084	186.695.924
Avg.	**0,285**	**57.939.429**	**60.569.867**	**Avg.**	**0,182**	**57.058.789**	**59.689.228**

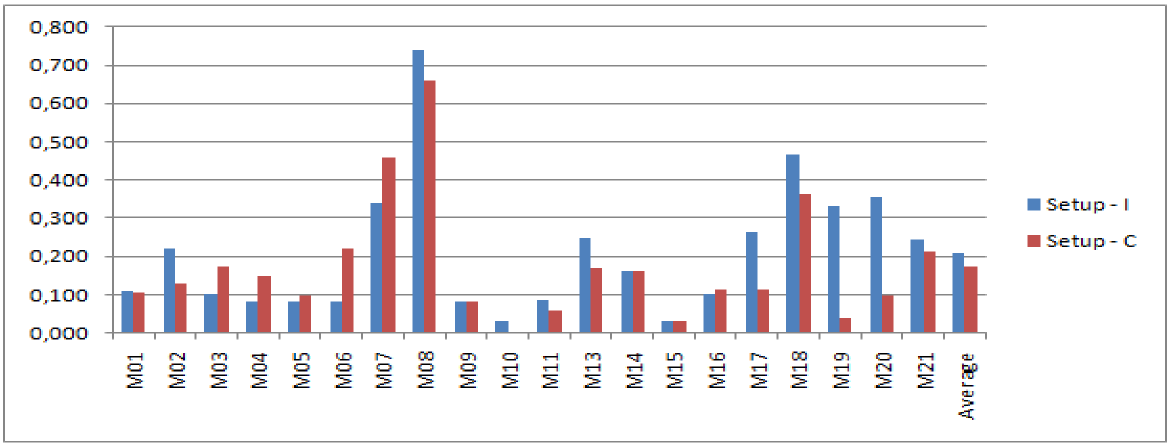

Figure 6.2. Initial – corrected setup comparison (Stage 1 – Set 2)

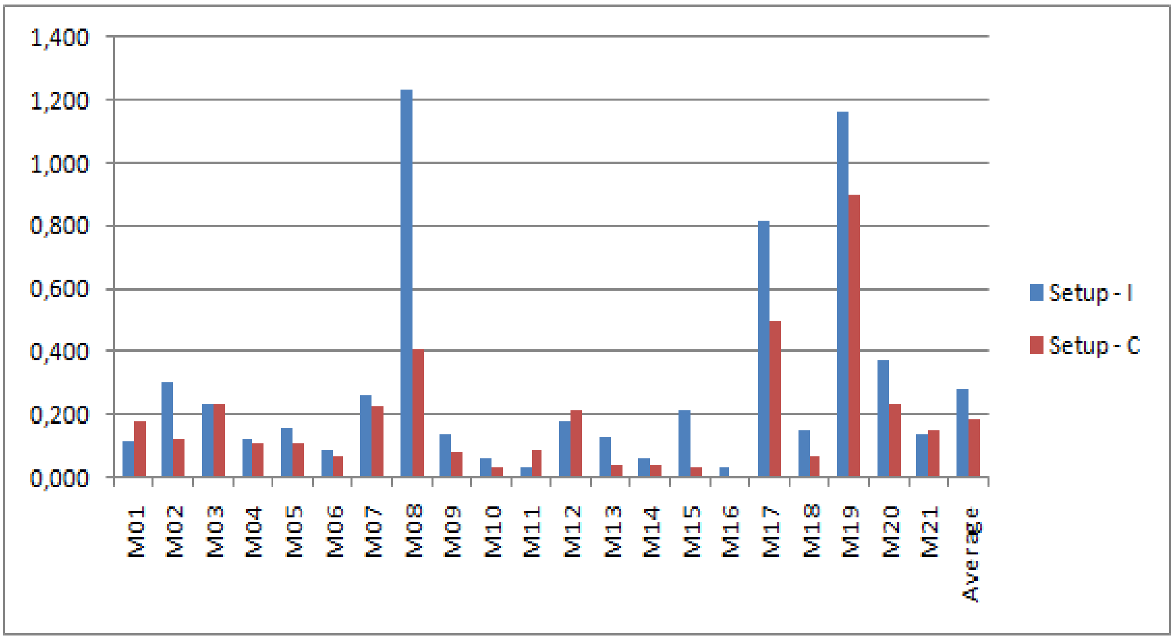

Figure 6.3. Initial – corrected setup comparison (Stage 1 – Set 3)

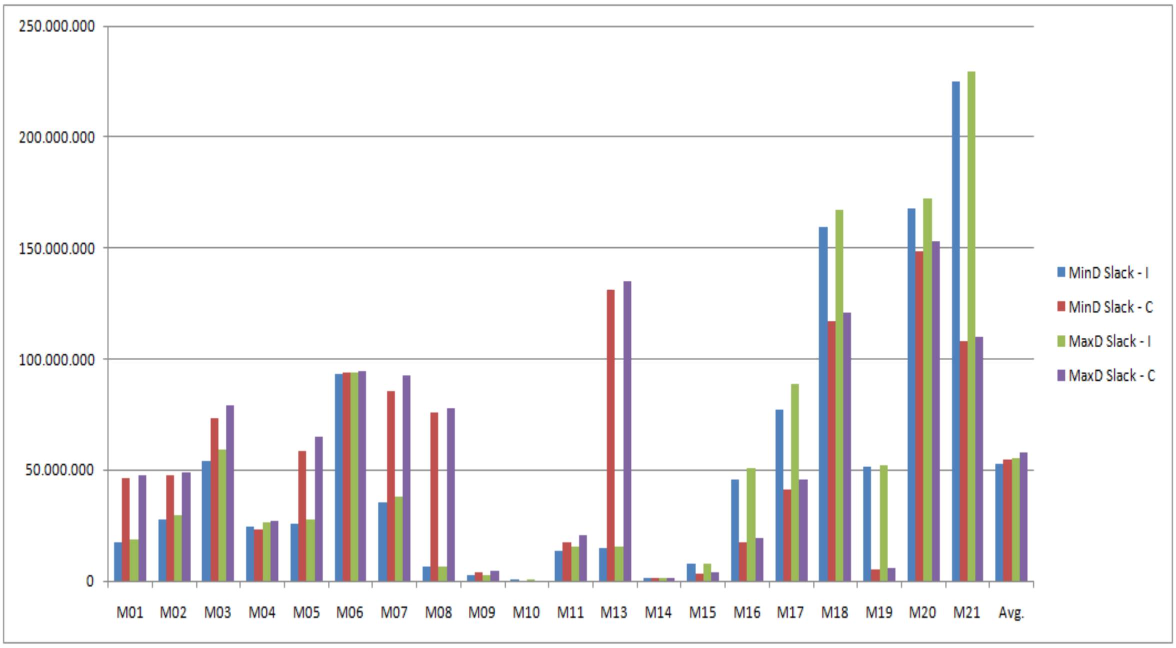

Figure 6.4. Initial – corrected slack comparison (Stage 1 – Set 2)

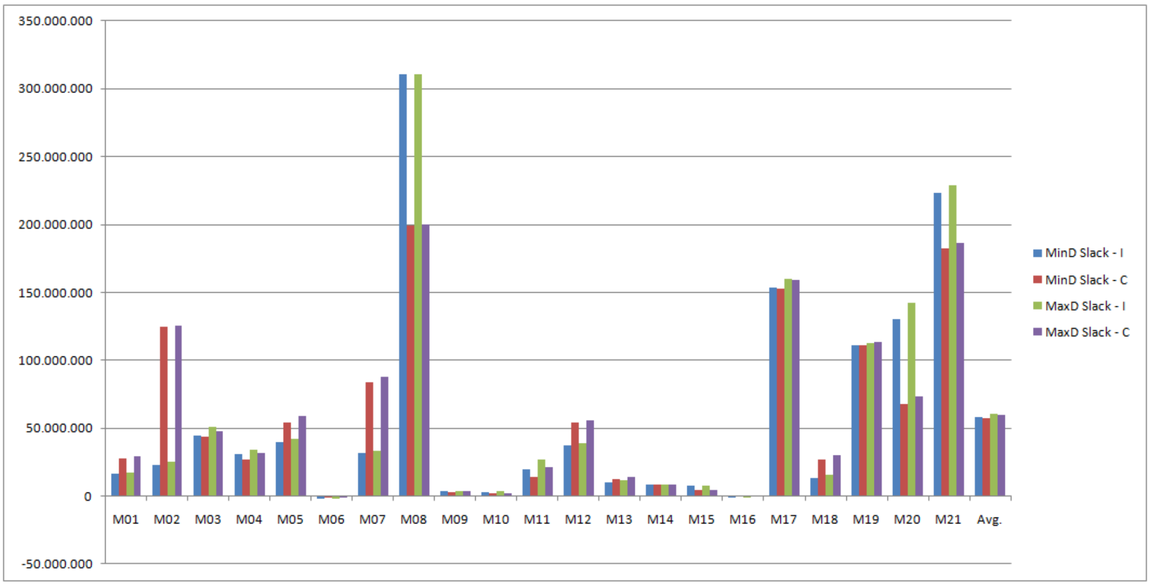

Figure 6.5. Initial – corrected slack comparison (Stage 1 – Set 3)

Tables and graphics above show the improvement in the result between initial solution and simulated annealing solution. When we compare setup costs, for both of the sets, set 2 and set 3, on average there is a reduction of 17% in set 2 (from 0,207 to 0,171) and 36% in set 3 (from 0,285 to 0,182). On the other hand, for data set 2 slack value is increased by approximately 5% from initial to final solution, which is also desirable and suitable with the objective. This increase is approximately same for both slack to the minimum due date and slack to the maximum due date parameters. For data set 3, this parameter decreases by 1,5%, which means it worsens. However, the gain from the setup side is much higher than the due date parameters so the overall result of the simulated annealing algorithm is successful for both of the data sets.

Simulated annealing iteration tests are also performed. During simulated annealing, the values of the parameters in objective function change. These values are recorded in the system and presented in Figure 6.6 and 6.7.

The most effective parameter is the location parameter as the objective function value moves simultaneously with this parameter. In general, all the sub parameters are changing in positive manner since an increase in minimum and maximum due date slack parameters makes positive effect on objective function. The value of the objective function sometimes increases showing worse solutions are accepted in those iterations. However, at the end, the value of the objective function is well below the point where the worse solutions are accepted, which shows the success of the heuristic.

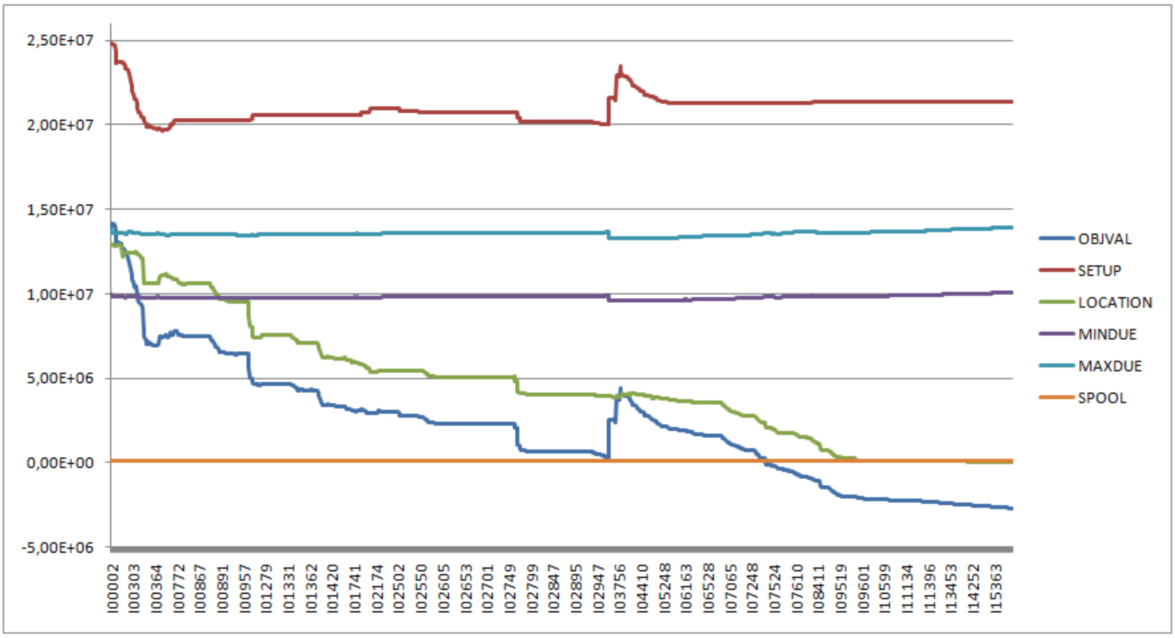

Figure 6.6. Simulated annealing iterations (Set 2)

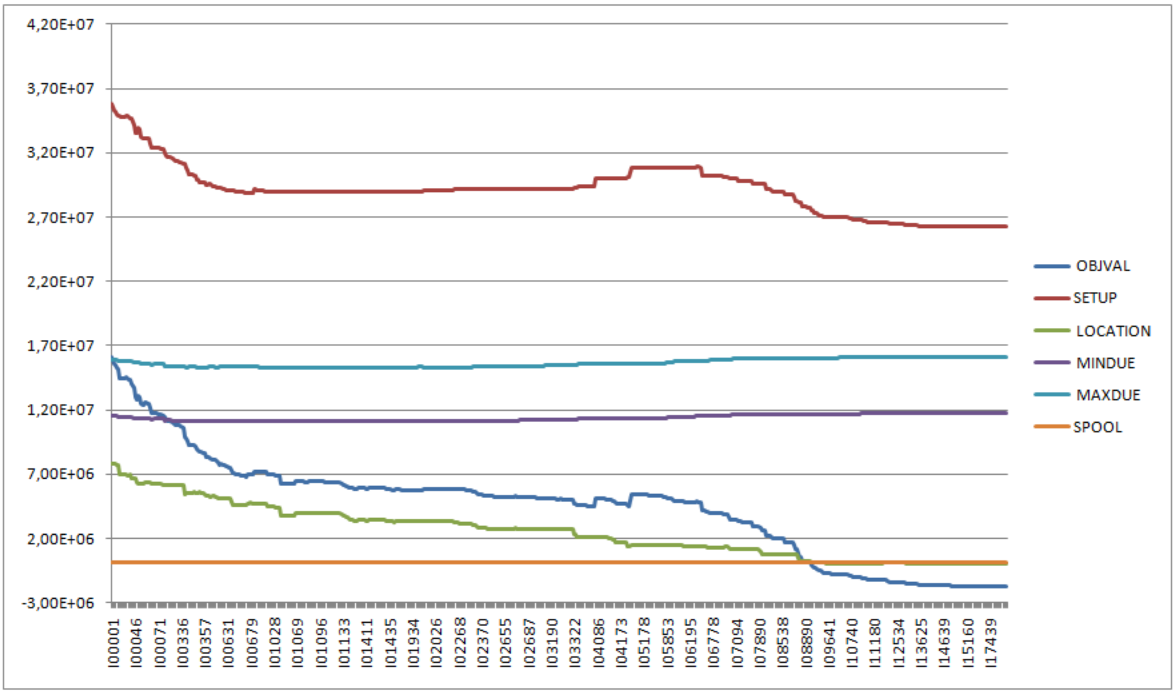

Figure 6.7. Simulated annealing iterations (Set 3)

Planners current solution and simulated annealing result are compared. When we compare the setup cost of the planner's solution and the initial solution, we realized that overall setup cost of the planner is much higher than the heuristics setup costs (Table 6.4.). One possible reason for this variance is that in Erbakır, planners intensively plans first 10 days of the planning horizon and do not consider the plan after first 10 days. As the planning window moves forward, then planners start to plan the production that enters the 10 day planning window. Due to this fact, when we limit the start date up to 10 days (which is 864.000 seconds), setup costs of the planners solution is better than the initial solution of the heuristic in data set 2. However, after simulated annealing, both data sets yields better solution than the intensively planned first 10 days in terms of setup costs.

Table 6.4. Setup cost comparison – planner vs. heuristic

	Planner Setup (Full Period)	Heuristic Setup (Full Period)	Planner Setup (Init. Period)	Heuristic Setup (Init. Period)	Heur. Setup (Init Period-Init Solution)
Set 2	10,403	3,425	4,029	3,369	4,146
Set 3	14,140	4,378	7,069	4,090	5,988

The overall result after stage 2 is compared with the planners result. The comparison is on Table 6.5.

Table 6.5. Final comparison of overall solution

	Location Change Penalty	Spooler / Comp. Count. Penalty	Due Date Penalty	Setup Penalty	Total Penalty
Set 2 – Planner	4,25E+05	1,43E+06	9,09E+05	9,36E+04	2,86E+06
Set 2 – Heuristic	5,10E+04	6,40E+05	1,41E+05	3,08E+04	8,63E+05
Set 3 – Planner	4,27E+05	1,40E+06	5,50E+05	1,27E+05	2,51E+06
Set 3 – Heuristic	5,90E+04	4,91E+05	1,43E+05	3,44E+04	7,28E+05

In general, the overall result after the second stage is better than the planner solution in the system in terms of every sub parameter of the objective function.

7. CONCLUSION

In this study, we focused on a scheduling problem in copper wire industry. The motivation behind the selection of this problem is the challenge of defining a scheduling problem in copper-wire industry which is not defined before with the sector specific complex constraints. We worked on a real life problem in our case study and the experience of the company's planners gave insight during the definition of the problem. The company in copper-wire industry that we have implemented the solution procedures is Erbakır Elektrolit.

At the beginning of the study, we summarized the general structure and process flow of a copper-wire production in order to see the big picture. Basic constraints are presented in this section and the focus area of the study is set by taking into consideration of the most complex and problematic sections.

Literature is reviewed extensively after the problem definition. Our problem can be categorized as hybrid flowshop in broadest term and we started the review in this area. Since our study is focusing on real life problem, we also focused on the papers about real life problems in flexible manufacturing systems. Then, specific problematic areas are investigated in the literature like parallel machines, sequence dependent setup times and costs. Different solution procedures are described in literature about these problems and we searched the most important and related ones. Finally, one of the most effective and suitable technique is chosen in our study, which is simulated annealing. Literature is also checked for simulated annealing solution procedures.

Once the focus area, multi-wire and bunched wire stages, is decided, the properties of the problem are defined. We defined fourteen main property of this scheduling problem at this step. These properties are presented with examples and illustrations to increase the level of understanding. Some of the properties can be ignored since the problems in those areas are not critical. In the assumptions part these are declared and the focus of the study is clearly defined afterwards.

After the definition of the problem and review of the literature, we represented the constraints and properties of the problem with a mathematical model. The properties are linked to the mathematical constraints and the objective function created in line with the objective of the business. Some of the non-linear constraints are converted to linear constraints with binary variables. On the other hand, one of the constraints remained as non-linear at this stage. For the further study, we can focus on the constraints and try to solve the same problem with a linear programming method.

Since the mathematical formulation contains high level of complexity, we focused on solving the planning problem with a heuristic approach. Our heuristic composed of four steps: data collection and general calculations, initial solution for multi-wire stage, corrective algorithm for the initial solution in multi-wire and a solution procedure in bunching production.

The steps of the heuristics are designed conceptually first and the pseudo code of the heuristics are defined. This flow is coded in the customer's live SAP SCM environment. The coding is done in ABAP language, SAP's internal programming language, and the parameters are designed as to be input for the developed program. The weight parameters are identified manually by discussing with the planners who knows the importance of each factor in the company. Three different data sets are collected from the live SCM system weekly. Also, the planner's production plan is downloaded to custom tables in order to compare the results and measure the efficiency of the heuristics.

The first step of the program calculates most used and changing parameters for future use. In the second step an initial solution for the first stage is obtained. This initial solution becomes input to the third step where the result of the initial solution is corrected via a search algorithm, which is simulated annealing. The run time parameters of the simulated annealing algorithm are obtained after several tests with one of the data sets. The definition of the parameters step is followed by running the simulated annealing algorithm to correct and improve the initial solution in the first step.

The result of the simulated annealing is given as input to the second stage heuristic. As the first stage is defined, parameters like spool type definition, component count are already given also. The result of the final step is compared with the planners result and our proposed heuristic outperforms the schedule on the planning system. The difference is high between our schedule and the schedule in the system. One explanatory reason for this might be that planners in the company mostly focus on the first 10 days of production. After 10 days, jobs are only scheduled without strict rules. This fact may explain the difference of the results between our study and current schedule.

For further study, the mathematical model can be solved via linear optimization techniques and the results can be compared with the heuristic proposed in this study. Also, this study is limited with only two data sets because of the fact that the duration of the study is not long enough and it is not easy to generate data from the real life environment. The heuristics can be tested with more data sets in order to get better planning solution. One other improvement to this study may be the comparison method between current and proposed solution. It is also possible to request from the planners to plan further periods also and we can download the planning result at that time to make the comparison healthier.

APPENDIX A: PARAMETER TUNING RESULTS

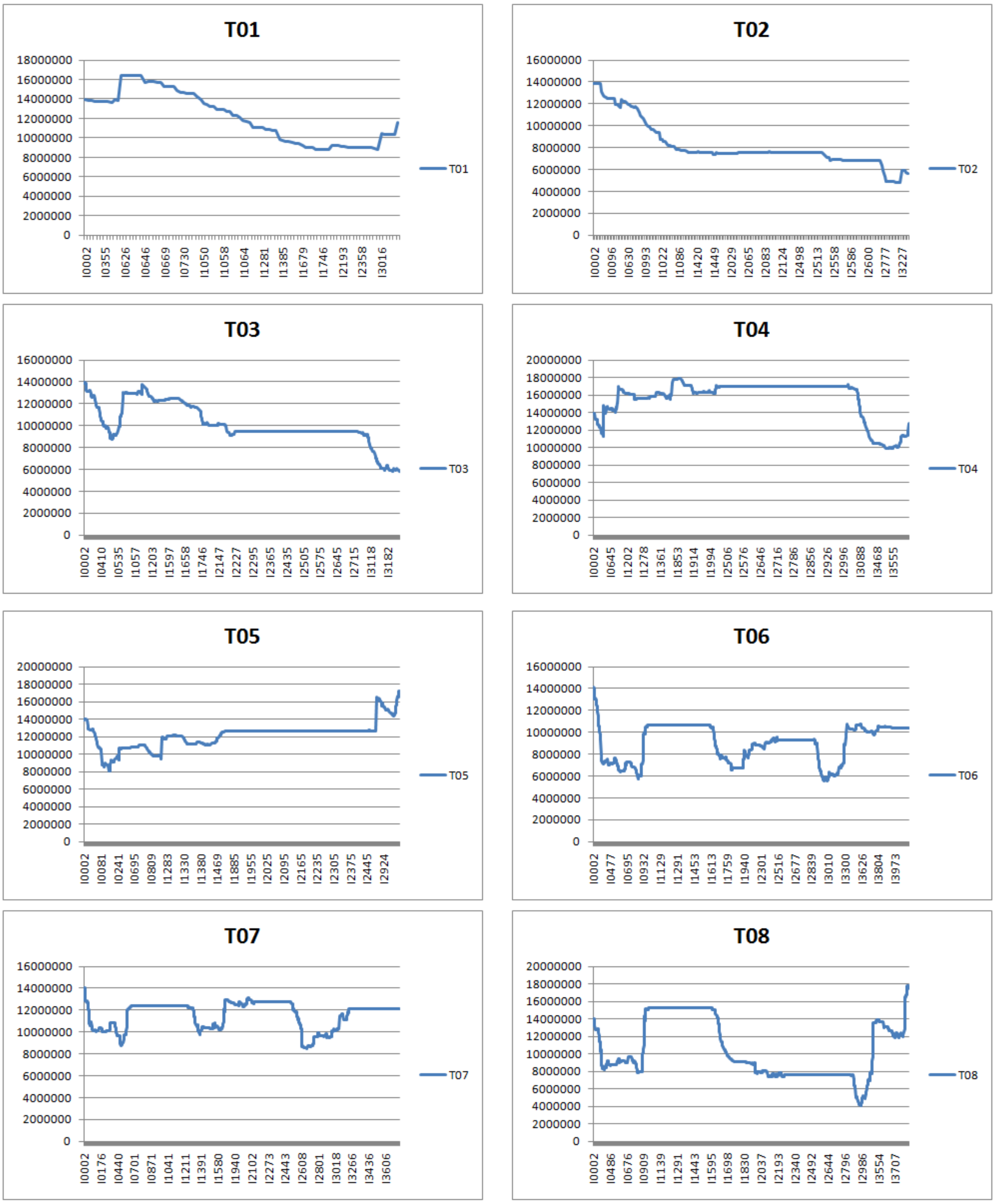

Figure A.1. Parameter tuning result graphics

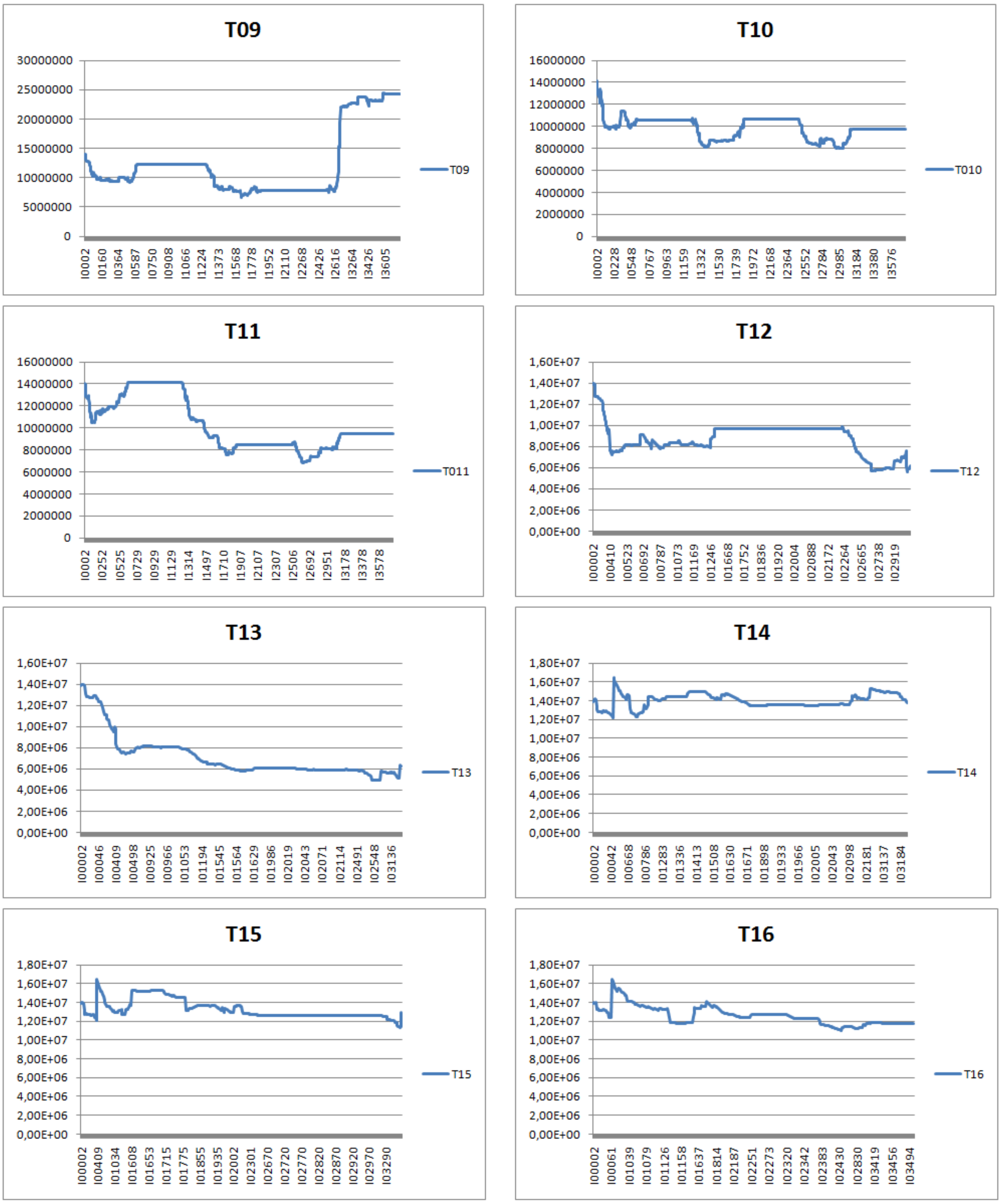

Figure A.1. Parameter tuning result graphics (continued)

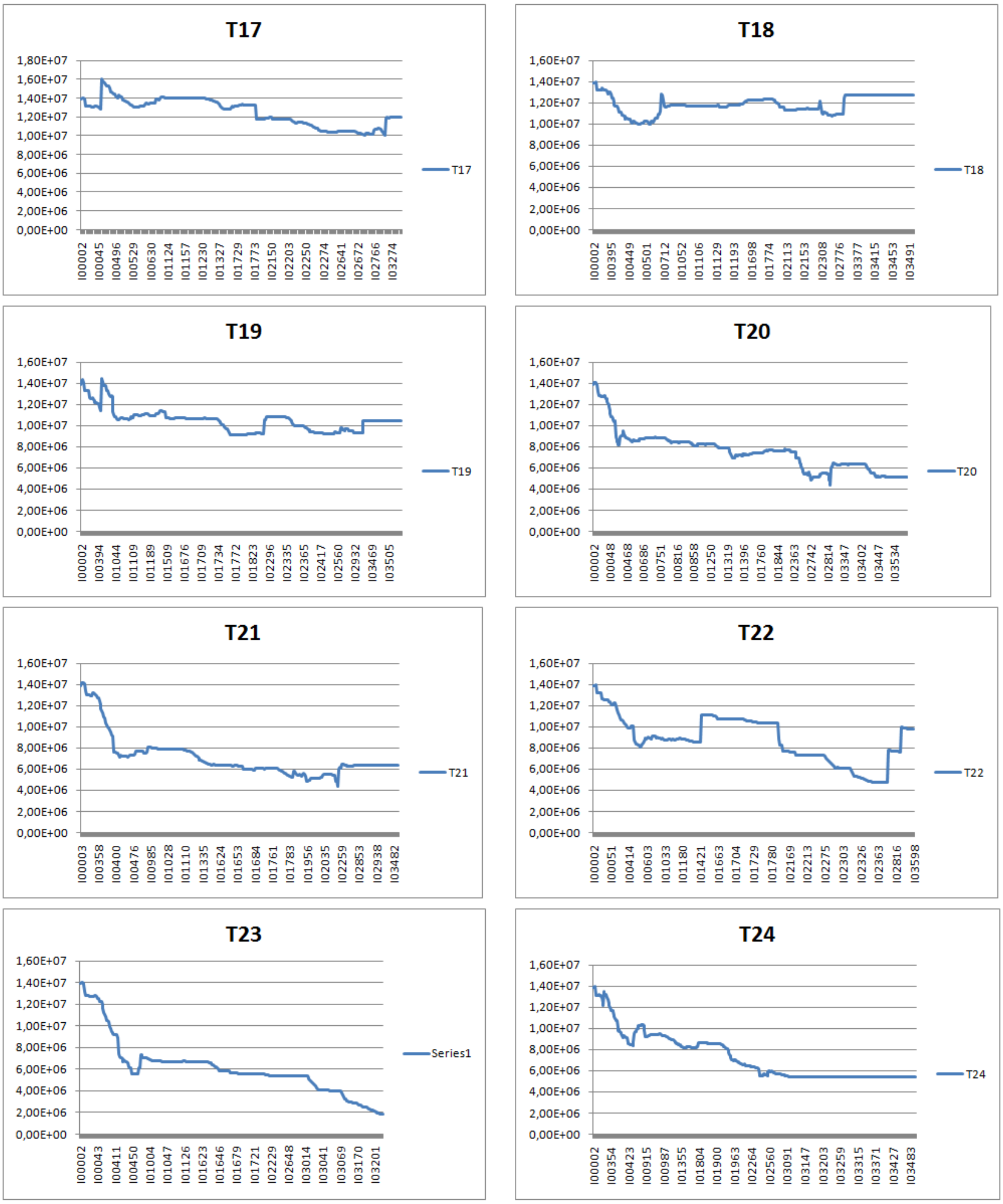

Figure A.1. Parameter tuning result graphics (continued)

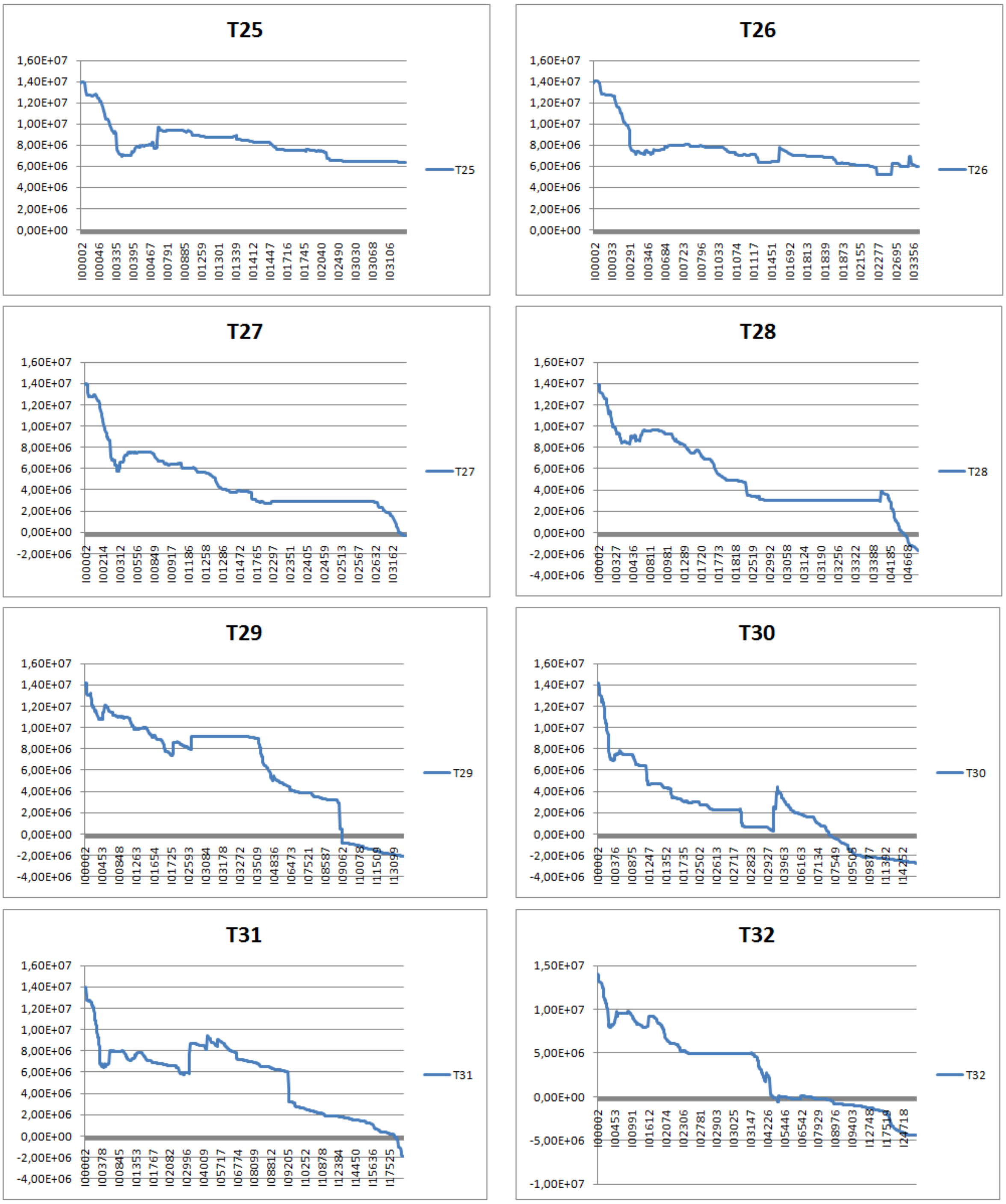

Figure A.1. Parameter tuning result graphics (continued)

REFERENCES

1. Linn, R and W. Zhang, "Hybrid flow shop scheduling: A survey", *Computers & Industrial Engineering*, Volume 37, Issues 1-2, pp. 57-61, October 1999.

2. Quadt, D., "Lot-sizing and scheduling for flexible flow lines", *Springer Science + Business Media*, New York, 2004.

3. Quadt, D. and H. Kuhn, "A taxonomy of flexible flow line scheduling procedures", *European Journal of Operational Research,* 178, pp. 686–698, 2007.

4. Gupta, J. N. D. and E. F. Stafford Jr., "Flowshop scheduling research after five decades", *European Journal of Operational Research,* 169, pp. 699–711, 2006.

5. Koulamas, C. and G. J. Kyparisis, "A modified LPT algorithm for the two uniform parallel machine makespan minimization problem", *European Journal of Operational Research,* 196, pp. 61–68, 2009.

6. Yalaoui, F. and C. Chu, "Parallel machine scheduling to minimize total tardiness", *Inernational Journal of Production Economics*, 76, pp. 265–279, 2002.

7. Lin, B.M.T. and A.A.K. Jeng, "Parallel-machine batch scheduling to minimize the maximum lateness and the number of tardy jobs", *International Journal of Production Economics,* 91, pp. 121-134, 2004.

8. Armentano, V. A. and M. F.de F. Filho, "Minimizing total tardiness in parallel machine scheduling with setup times: An adaptive memory-based GRASP approach", *European Journal of Operational Research*, 183, pp. 100–114, 2007.

9. Ruiz, R. and C. Maroto, "A genetic algorithm for hybrid flowshops with sequence dependent setup times and machine eligibility", *European Journal of Operational Research*, 169, pp. 781-800, 2006.

10. Chen, C. and C. Chen, “Bottleneck-based heuristics to minimize total tardiness for the flexible flow line with unrelated parallel machines”, *Computers & Industrial Engineering,* 56, pp. 1393–1401, 2009.

11. Chen, C. and C. Chen, “A bottleneck-based heuristic for minimizing makespan in a flexible flow line with unrelated parallel machines”, *Computers & Operations Research,* 36, pp. 3073 – 3081, 2009

12. Bellanger, A. and A. Oulamara, “Scheduling hybrid flowshop with parallel batching machines and compatibilities”, *Computers & Operations Research*, 36, pp. 1982 – 1992, 2009.

13. Yaurima, V., L. Burtseva and A. Tchernykh, “Hybrid flowshop with unrelated machines, sequence-dependent setup time, availability constraints and limited buffers”, *Computers & Industrial Engineering*, 56, pp. 1452–1463, 2009.

14. Allaoui H. and A. Artiba, “Scheduling two-stage hybrid flow shop with availability constraints”, *Computers & Operations Research,* 33, pp. 1399–1419, 2006.

15. Lowa C., C. Hsub and C. Sub, “A two-stage hybrid flowshop scheduling problem with a function constraint and unrelated alternative machines”, *Computers & Operations Research,* 35, pp. 845 – 853, 2008.

16. Allahverdi A., C.T. Ng, T.C.E. Cheng and M. Kovalyov, “A survey of scheduling problems with setup times or costs”, *European Journal of Operational Research,* 187, pp. 985–1032, 2008.

17. Allahverdi, A, J.N.D. Gupta and T. Aldowaisan, “A review of scheduling research involving setup considerations”. *The International Journal of Management Sciences,* 27, pp. 219–239, 1997.

18. Allahverdia, A., and F. S.Al-Anzib, “The two-stage assembly scheduling problem to minimize total completion time with setup times”, *Computers & Operations Research*, 36, pp. 2740 – 2747, 2009.

19. Ferreira, D., R. Morabito and S. Rangel, “Solution approaches for the soft drink integrated production lot sizing and scheduling problem”, *European Journal of Operational Research*, 196, pp. 697–706, 2009.

20. Voβ, S. and A. Wittb, “Hybrid flow shop scheduling as a multi-mode multi-project scheduling problem with batching requirements: A real-world application”, *International Journal of Production Economics,* 105, pp. 445–458, 2007.

21. Naderi, B., R. Ruiz and M. Zandieh, “Algorithms for a realistic variant of flowshop scheduling”, *Computers & Operations Research*, doi:10.1016/j.cor.2009.04.017, 2009.

22. Zandieh, M. and F. Jabbari, “Hybrid flexible flowshops with sequence-dependent setup times and machine availability constraints”, *Computers & Industrial Engineering,* doi: 10.1016/j.cie.2009.03.012, 2009.

23. Bagchi, T. P., J. N. D. Gupta and C. Sriskandarajah, “A review of TSP based approaches for flowshop scheduling”, *European Journal of Operational Research,* 169, pp. 816–854, 2006.

24. Nishi, T., Y. Hiranaka and M. Inuiguchi, “Lagrangian relaxation with cut generation for hybrid flowshop scheduling problems tominimize the total weighted tardiness”, *Computers & Operations Research*, doi:10.1016/j.cor.2009.04.008, 2009.

25. Zhenbo, W. and X. Wenxun, “Parallel Machine Scheduling with Special Jobs”, *Tsinghua Science and Technology*, Volume 11, Number 1, pp. 107-110, February 2006.

26. Amin-Naseri M. R. and M. A. Beheshti-Nia, "Hybrid flow shop scheduling with parallel batching", *International Journal of Production Economics*, Volume 117, Issue 1, pp. 185-196, January 2009.

27. Laub, J. D., J. W. Fowler and A. B. Keha, "Minimizing makespan with multiple-orders-per-job in a two-machine flowshop", *European Journal of Operational Research,* 182, pp. 63–79, 2007.

28. Toktaş, B., M. Azizoğlu and S. K. Köksalan, "Two-machine flow shop scheduling with two criteria: Maximum earliness and makespan", *European Journal of Operational Research* 157,pp. 286–295, 2004.

29. Sarin, S. C and P. Jaiprakash, "Flow shop lot streaming", Springer Science + Business Media, New York, 2007.

30. Langevin A., D. Riopel and K. E. Stecke, "Transfer batch sizing in flexible manufacturing Systems", *Journal of Manufacturing Systems,* Vol. 18/No. 2, 1999

31. Sriskandarajah C. and E. Wagneur, "Lot streaming and scheduling multiple products in two-machine no-wait flowshops", *IIE Transactions,* 31, pp. 695-707, 1997.

32. Chen, J. and G. Steiner, "Lot streaming with detached setups in three-machine flow shops", *European Journal of Operational Research*, 96, pp. 591-611, 1996.

33. Kalir, A. A. and S. C. Sarin, "Constructing near optimal schedules for the flow-shop lot streaming problem with sublot-attached setups", *Journal of Combinatorial Optimization,* 7, pp. 23–44, 2003.

34. Huang, R. and C. Yang, "Overlapping production scheduling planning with multiple objectives—An ant colony approach", *International Journal of Production Economics,* 115, pp. 163– 170, 2008.

35. Martin, C. H., "A hybrid genetic algorithm/mathematical programming approach to the multi-family flowshop scheduling problem with lot streaming*", Omega*, 37, pp. 126 – 137, 2009.

36. Logendran, R., B. McDonell and B.Smucker, Scheduling unrelated parallel machines with sequence-dependent setupset, *Computers & Operations Research,* 34, pp. 3420-3438, 2007.

37. Kim, D., K. Kim, W. Jang and F. F. Chen, "Unrelated parallel machine scheduling with setup times using simulated annealing", *Robotics and Computer Integrated Manufacturing*, 18, pp. 223-231, 2002.

38. Lee, Y. H. and M. Pinedo, "Scheduling jobs on parallel machines with sequence-dependent setup times", *European Joumal of Operational Research*, 100, pp. 464-474, 1997.

39. Weng, M. X., J. Lu and H. Ren, "Unrelated parallel machine scheduling with setup consideration and a total weighted completion time objective*", International Journal of Production Economics,* 70, pp. 215-226, 2001.

40. Kirkpatric S., C.D. Gelatt and M.P. Vecci, "Optimization by simulated annealing", *Science,* 220, pp. 671–680, 1983.