ROBOTIC ASSEMBLY LINE DESIGN WITH TOOL CHANGES

A THESIS

SUBMITTED TO THE DEPARTMENT OF INDUSTRIAL ENGINEERING

AND THE INSTITUTE OF ENGINEERING AND SCIENCE

OF BILKENT UNIVERSITY

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR THE DEGREE OF

MASTER OF SCIENCE

By

Adnan Tula

July, 2009

ABSTRACT

ROBOTIC ASSEMBLY LINE DESIGN WITH TOOL CHANGES

Adnan Tula
M.S. in Industrial Engineering
Supervisor: Prof. Dr. M. Selim Aktürk
July, 2009

This thesis is focused on assembly line design problems in robotic cells. The mixed-model assembly line design problem that we study has several subproblems such as allocating operations to the stations in the robotic cell and satisfying the demand and cycle time within a desired interval for each model to be produced. We also ensure that assignability, precedence and tool life constraints are met. The existing studies in the literature overlook the limited lives of tools that are used for production in the assembly lines. Furthermore, the studies in the literature do not consider the unavailability periods of the assembly lines and assume that assembly lines work 24 hours a day continuously. In this study, we consider limited lives for the tools and hence we handle tool change decisions. In order to reflect a more realistic production environment, we deal with designing a mixed-model assembly line that works 24 hours a day in three 8-hour shifts and we consider lunch and tea breaks that are present in each shift. This study is the first one to propose using such breaks as tool change periods and hence eliminate tool change related line stoppages. In this setting, we determine the number of stations, operation allocations and tool change decisions jointly. We provide a heuristic algorithm for our problem and test the performances of our heuristic algorithm and DICOPT and CPLEX solvers included in GAMS software on different instances with varying problem parameters.

Keywords: Robotic cell, assembly line, tool change, heuristic algorithm.

ÖZET

UÇ DEĞİŞİMLİ ROBOTİK MONTAJ HATTI TASARIMI

Adnan Tula
Endüstri Mühendisliği, Yüksek Lisans
Tez Yöneticisi: Prof. Dr. M. Selim Aktürk
Temmuz, 2009

Bu tezin konusu robotik hücrelerde montaj hattı tasarım problemleridir. Çalıştığımız montaj hattı tasarım problemi operasyonların istasyonlara atanması ve üretilecek her model için talep ve çevrim zamanının belli aralıklar içinde karşılanması gibi alt problemler içermektedir. Bunun yanı sıra atanabilirlik, öncelik ve uç ömrü kısıtları sağlanmaktadır. Literatürde var olan çalışmalarda, montaj hatlarında üretim için kullanılan uçların kısıtlı ömrü olduğu gözardı edilmiştir. Ayrıca, literatürde yer alan çalışmalar 24 saat kesintisiz üretimi temel almakta ve montaj hatlarında üretim yapılamayan zamanlar hesaba katılmamaktadır. Bu tezde, kısıtlı uç ömürleri kullanılmakta, dolayısıyla uç değişim kararları da ele alınmaktadır. Daha gerçekçi bir üretim ortamı yansıtmak amacıyla, karma model üretimi yapılan, içinde çay ve yemek molalarının olduğu 8 saatlik üç vardiya düzeniyle günde 24 saat çalışan bir montaj hattı tasarım problemi ele alınmıştır. Bu çalışma, çay ve yemek molalarının uç değiştirme zamanı olarak kullanılmasını ve bunun sonucunda uç değişim zorunluluğundan kaynaklanan üretim hattı durdurmalarının önlenmesini önermesi açısından ilktir. Bu bağlamda, istasyon sayıları, operasyonların istasyonlara atanması ve uç değişim kararları birlikte ele alınmaktadır. Çalışılan problem için çözüm yolu olarak bir sezgisel algoritma geliştirilmiş, değişken değerlerin atandığı parametrelerle oluşturulan değişik örnekler üzerinde sezgisel algoritma ile DICOPT ve CPLEX çözücülerinin performansları karşılaştırmalı olarak test edilmiştir.

Anahtar sözcükler: Robotik hücre, montaj hattı, uç değişimi, sezgisel algoritma.

To my father...

Acknowledgement

First and foremost, I would like to express my gratitude to my advisor, Prof. Dr. M. Selim Aktürk, for his invaluable guidance and helps during my M.S. study. With his support and advices, he has always been more than an advisor to me.

I am also grateful to Prof. Dr. Erdal Erel and Assoc. Prof. Dr. Oya Ekin Karaşan for accepting to read and review this thesis and for their invaluable suggestions.

I would like to express my deepest gratitude to my father, Hüseyin Tula and my mother, Necla Tula for their endless love, encouragement and support. I have always felt very lucky that I am their child, and I will always do my best to keep them feeling the same for me.

I am indebted to all my friends who were always with me and made my life more beautiful. I am especially grateful to İhsan Yanıkoğlu, Utku Guruşçu, Könül Bayramoğlu, Safa Onur Bingöl, Merve Çelen, Onur Özkök, Sibel Alumur, Duygu Tutal, Ezel Ezgi Budak and Ceyda Kırıkçı for everything they have done for me.

Finally, I would like to express my special thanks to TÜBİTAK for the scholarship provided throughout the thesis study. The research was partially supported by Tofaş Türk Otomobil Fabrikası A.Ş. (Fiat Turkey) as Bilkent University Project No: 300 189 2 5.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

An important application of robots in the automotive industry is their use for spot welding operations in robotic cells. Once an automotive component of a vehicle is designed and the required spot welds to assemble the pieces on this component are determined, the problem of allocating the welding operations to robotic cell stations arises. Different allocations result in different production quantities (e.g., cycle times), station investment costs and tooling costs that are incurred by replacing the tools with new ones. Each station consists of a single robot in a fully automated production line. In order to perform welding operations, the robots use spot welding guns that we refer to as welding tools throughout this study. A welding tool has a limited life time that is represented by the total number of spot welds that it can process.

This study focuses on a robotic cell mixed-model assembly line design problem in which multiple models can be produced in any order. Our objective is to maximize the total profit that will be gained from the assembly line by manufacturing automotive body components. The problem has several subproblems which include allocation of welding operations to the stations, satisfying the demand and cycle time within a desired interval for the parts to be produced. While defining the problem and constraints, we were inspired by a real-life assembly line problem we faced in a project that we formerly conducted at one of the leading companies in the automotive industry in Turkey.

Today, improvements in technology and automation and large investment costs incurred for building assembly lines have increased the importance of studies on designing efficient assembly lines. Assembly line designing problems attract much attention from the academic world. Therefore, numerous studies have been conducted by the researchers, in which some aspects such as equipment selection and balancing of the lines are considered.

1.1 Literature Review

As described by Baybars [3], there are two main groups of assembly line studies. Simple assembly line balancing problems (SALBP) consider a single product to be produced in the assembly lines. In the literature, the widely used objectives for these studies include minimizing the number of workstations or the cost for building assembly lines for a given cycle time and minimizing the cycle time for a given number of workstations. There are many studies in the field of SALB, some of which are Baybars [2], McMullen and Tarasewich [16], Fleszar and Hindi [10], Rekiek *et al.* [19], Scholl and Klein [22] and Levitin *et al.* [14]. For a review of studies in the SALB field, we can refer to the review paper of Scholl and Becker [21].

In general assembly line balancing problems (GALBP), SALB problems are extended by introducing other aspects such as mixed-model or multi-model cases, zoning constraints and parallel stations. As described by Becker and Scholl [4], in a mixed-model line, different models are produced in the assembly line in an arbitrary inter-mixed sequence. There are several studies in the literature for the version of assembly line balancing problem with mixed-model lines. Bukchin *et al.* [5] address the problem of designing mixed-model assembly lines in which a make-to-order policy is followed. By separating the assembly tasks into two sets, they propose a three-stage heuristic in order to find a solution with the minimum number of stations for a given cycle time: in the first stage, they assign the first set of tasks that should be assigned to the same station for each model requiring these tasks. Then, they assign the second set of tasks that can be assigned to different stations for different models to balance each model in the assembly line, subject to the constraints resulting from the assignment of the first set of tasks. The final stage is to improve the solution by a neighborhood search. Erel and Gökcen [7] use a shortest-route formulation to solve a mixed-model assembly line problem. Common assembly tasks between different models are assigned to the same station. They transform the problem into a single-model version by constructing a combined precedence diagram from the precedence diagrams of all models. Bukchin and Rabinowitch [6] relax the assumption of restricting a task that is common for several models to a particular station and allow such a task to be assigned to different stations for different models. They provide an integer formulation for their model and give a lower bound for the total station cost and task assignment costs, which they aim to minimize. In order to find optimal or near-optimal solutions, they propose a heuristic algorithm based on branch-and-bound. Haq *et al.* [12] present a hybrid genetic algorithm for the mixed-model assembly line balancing problem. They use the modified ranked positional weight method to obtain an initial assignment of tasks to stations. They reduce the search space and search time of their hybrid genetic algorithm by providing this initial solution to their algorithm. Different from the studies that adapt a single station at each stage of the assembly line, Askin and Zhou [1] proposed a nonlinear integer program for assigning tasks to stations in a serial line in which the stages of the assembly line consist of an arbitrary number of identical, parallel workstations. While considering precedence relations between tasks, they allow tooling selections for workstations and each task requires a certain type of tool. Their objective is to minimize the sum of the total fixed cost for operating stations and the total equipment/tooling cost. Because of the complexity of the problem, they provide an assignment heuristic for finding good initial solutions.

To address the problem of handling model changes in mixed-model lines, Matanachai and Yano [15] approach the mixed-model line problem with the objective of facilitating the construction of good sequences and short-term workload stability while assigning tasks to stations. They assume predetermined cycle times and number of stations and include additional terms in their objective function to minimize within- and between-station processing time diversity. Maintaining short-term workload stability provides robustness for daily model changes in mixed-model lines. Merengo *et al.* [17] present balancing and sequencing methodologies for mixed-model assembly lines to minimize the number of workstations for predetermined model cycle times, reduce work-in-process and minimize the rate of incomplete jobs. By presenting integer programming formulations and providing numerical results, Sawik [20] compared monolithic and hierarchical balancing and sequencing approaches in which balancing and sequencing of mixed-model lines are determined simultaneously and sequencing of models proceeds the balancing of the line, respectively.

Sparling and Miltenburg [23] were the first to study the mixed-model U-line balancing problem to minimize the number of stations in the assembly line. For this problem, they presented a four-step approximate solution algorithm: the first two steps transform the mixed-model problem into an equivalent single-model problem, the third step finds the optimal workload balance for the equivalent single-model problem and the final step transforms the balance found in the third step into a feasible balance for the original mixed-model problem. Miltenburg [18] extended balancing and sequencing problem to U-shape mixed-model lines in a just-in-time environment. A nonlinear mixed-integer programming formulation is presented to solve the balancing and sequencing problem simultaneously. Erel *et al.* [8] developed a simulated annealing-based algorithm to minimize the number of stations in a U-type assembly line and tested the performance of their proposed algorithm against optimum seeking DP and IP-based algorithms and other heuristic procedures. In contrast to deterministic processing times, Erel *et al.* [9] studied U-line balancing problem with stochastic task times. Their study was the first one to propose a beam search-based method to minimize total expected cost, which consists of total labor cost and total expected incompletion cost. Van Hop [13] addressed the mixed-model assembly line problem with fuzzy processing times for the first time. A heuristic to aggregate fuzzy processing times is proposed and the problem is transformed into a a single-model version by using a combined precedence diagram. A heuristic is also developed to find a solution for the fuzzy assembly line problem. Vilarinho and Simaria [24] address some additional zoning constraints in their study for the assignment of tasks to stations. They assume a subset of tasks that are linked and must be assigned to the same station and subsets of incompatible tasks that should be assigned to different stations. They also assume that some tasks can only be assigned to particular stations. They allow parallel stations in the assembly line and present an ant colony optimization algorithm for minimizing the number workstations for a given cycle time. Wilhelm and Gadidov [25] devised a branch-and-cut approach to minimize the total assembly line cost that includes station activating, machining and tooling costs considering machine capacities and available tool spaces in the stations. Each operation requires a set of tools to be performed so tooling requirements constitute an important part of the problem. Machine capacity is used as an analog of cycle time restriction that appears in single-model assembly lines.

The studies in the literature present different model formulations and propose different solution techniques for assembly line problems with different objective functions. However, these studies do not consider the unavailability periods of the assembly lines and hence are based on continuous production. In addition to this, they overlook the limited lives of the tools that are used in production. As we will further discuss in Chapter 5, assembly lines may suffer from tool change related line stoppages unless the limited tool lives are taken into consideration while allocating the operations to the stations. Therefore, we consider these two important attributes of the robotic assembly lines in addition to other well-known constraints of the assembly line balancing problem.

1.2 Thesis Overview

Traditional assembly line design studies are generally based on the objectives of minimizing the cycle time, the number of stations and costs for building assembly lines or maximizing the efficiency of the assembly lines. In this study, different from the studies in the literature, we address a mixed-model assembly line problem with a profit maximization objective. We maximize the total profit function, which is the difference between the revenue gained by manufacturing components of the final products and the sum of the station investment costs and tooling costs. Station investment costs include robot cost, fixture cost and space cost and tooling costs are incurred by replacing the tools with new ones. In addition to this, the studies in the literature do not consider the unavailability periods of the assembly lines and assume that assembly lines work 24 hours a day continuously. However, we deal with designing a mixed-model assembly line that works 24 hours a day in three 8-hour shifts. We consider lunch and tea breaks to reflect a more realistic production environment. In the literature, there are many studies regarding tool selection and tool costs, but these studies ignore limited tool life and incorporate the assumption that tool change is omitted throughout the planning horizon and tooling costs are incurred once at the beginning of the production stage. Tool life has two implications. First, tools must be changed at the end of their tool lives and this will increase the tooling cost. Second, tool changes may correspond to a time when the assembly line is supposed to be operating and therefore may result in line stoppages. In our study, we consider that tools have limited lives that are represented by the total number of spot welds that they can perform. We use the lunch and tea breaks as tool change periods by which we aim to eliminate tool change related line stoppages. At each break, if there exist welding tools that have a remaining number of spots less than the total number of spots that they have to perform until the next break, they should be changed with new ones. Therefore, the tooling cost term that appears in our objective function is a function of the total number of tools used throughout the planning horizon.

The organization of this study is as follows: In the next chapter we will present a nonlinear mixed-integer formulation of our assembly line design problem. In Chapter 3, we will provide the linearized mixed-integer version of the same problem. In Chapter 4, a heuristic algorithm for finding good feasible solutions will be developed and this heuristic algorithm will be improved by a surrogate problem. An implementation of our study to a leading automotive company in Turkey will be provided in Chapter 5. Numerical results and several comparisons between our heuristic algorithm and GAMS software are given in Chapter 6. Concluding remarks are given in Chapter 7. A summary of all the problem parameters and decision variables is provided in the appendix.

Chapter 2

Problem Definition

In this chapter, we give the definition of our problem and introduce the parameters, variables and the mathematical model we will use to solve our problem.

We consider a robotic cell that contains at most m stations: $S_1, S_2, \ldots, S_m$. Let M={1, 2,..., m} be the set of indices of these stations. Space restrictions in the production area and a limited budget for investment costs are among the reasons of such a restriction on the number of stations. We use the parameter V_j to denote the cost of setting up station j, $j \in M$. The cost of setting up a station consists of robot cost, fixture cost and space cost. There are g different models of parts to be produced in this robotic cell and G={1, 2,..., g} is the index set of part models. O_{hi} represents operation i of model h, $i \in N_h$, $h \in G$, where N_h={1, 2,..., n_h} is the index set of welding operations of model h to be allocated to stations and n_h is the number of operations to be performed to produce a part of model h. Welding operations consist of a number of spot welds. Let W_{hi} be the number of spot welds required to perform operation i of model h, $i \in N_h$, $h \in G$. An operation may require a single spot weld. However, some operations require more than one spot weld in order to assemble the part at its proper geometry. In general, the spot welds that are close to each other with respect to their locations on the component are grouped together as operations. A welding tool has a limited lifetime that is represented by the total number of spot welds it can process. We define B_j as the total number of spot welds such that the welding tool in station $j \in M$ can process.

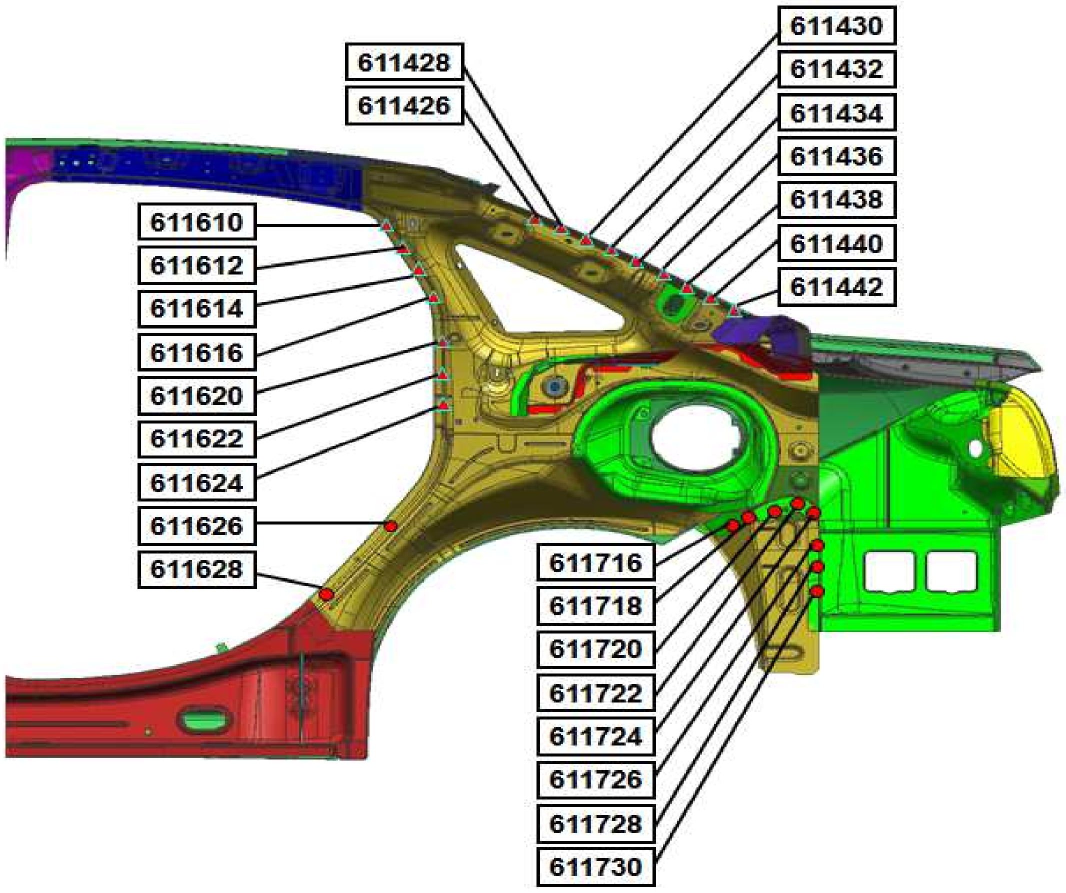


Figure 2.1: A spot weld scheme of an automotive body component

An example of an automotive body component is given in Figure 2.1. Figure 2.1 shows the spot welds required to perform some of the welding operations on different locations of a part. The 26 spot welds seen in this figure constitute a subset of all operations required to produce this body component. As we can see, a spot weld can be close to some spot welds but can also be distant from some other spot welds. Therefore, the locations of the spot welds on a part along with the minimum number of required spot welds to assemble the subcomponents to maintain their proper geometry during the part transfer between stations play an important role in grouping these spot welds as operations.

In order to set up a profitable assembly line and determine the number of stations required in the assembly line, we have to take the expected demand into consideration. We represent the yearly expected demand for the parts to be produced as θ_h, $h \in G$. We use the parameter γ_h to denote the target cycle time for model h to meet the yearly expected demand θ_h. In general, let $\bar{T}_h$ be the time allocated to production of model h. Let $\bar{f}_h$ be the cycle time of model h and let $\bar{\theta}_h$ be the corresponding production amount for model h. Then, the relationship between $\bar{f}_h$ and $\bar{\theta}_h$ is as follows:

$$\bar{\theta}_h = \frac{\bar{T}_h}{\bar{f}_h} \qquad \forall h.$$

We use the parameters γ_h^L and γ_h^U as the lower and upper bounds for the actual cycle time of model h, respectively. We neither accept to produce an amount of parts of model h less than the amount that can be produced when the actual cycle time is equal to γ_h^U nor can make any additional profit if production of model h exceeds the amount that can be produced when the actual cycle time is equal to γ_h^L. Up to the production amount of θ_h we assume a constant profit for each part of model h produced and denote it by PR_h. In case of producing between θ_h and the amount that can be produced when the actual cycle time is equal to γ_h^L, each excess part of model h produced contributes an expected profit denoted by PR_h^ϵ. If the production of model h exceeds the amount that can be produced when the actual cycle time is equal to γ_h^L, any excess part does not contribute any additional profit. We represent the relationship between PR_h and PR_h^ϵ as the following:

$$PR_h^\epsilon = PR_h - \Delta_h, \qquad \Delta_h \geq 0.$$

Let f_h be the cycle time of model h for a particular allocation. Let $\widehat{\theta}_h$ and θ_h^L be the production amounts of model h when the cycle time of model h is equal to the decision variable, f_h, and the given parameter, γ_h^L, respectively. In this setting, revenue contribution of each model h, which is the sum of the individual profits that each product of model h contributes, is calculated as in the following:

$$\text{Total Revenue} = \begin{cases} \widehat{\theta}_h \cdot PR_h & \text{if } \gamma_h \leq f_h \leq \gamma_h^U, \\ \theta_h \cdot PR_h + (\widehat{\theta}_h - \theta_h) \cdot PR_h^\epsilon & \text{if } \gamma_h^L \leq f_h \leq \gamma_h, \\ \theta_h \cdot PR_h + (\theta_h^L - \theta_h) \cdot PR_h^\epsilon & \text{if } f_h < \gamma_h^L. \end{cases}$$

Figure 2.2 shows an example of the relationship between the revenue contribution such that $PR_h = \$20$ and $PR_h^{\epsilon} = \$5$ and the cycle time for a particular model h that is produced in the assembly line, where $\gamma_h^U = 84$ seconds, $\gamma_h = 72$ seconds and $\gamma_h^L = 64$ seconds.

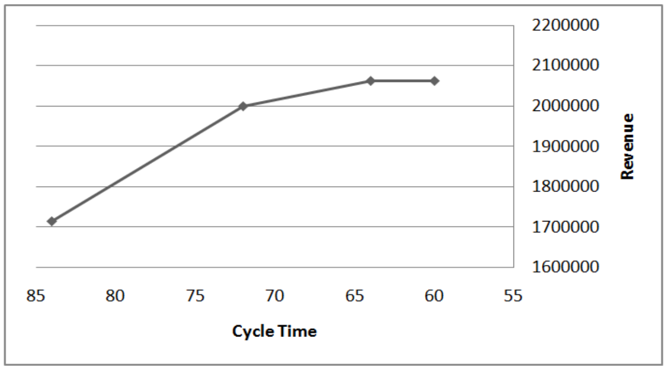


Figure 2.2: Cycle time - revenue contribution relationship

As we see in Figure 2.2, while the cycle time of model h decreases from γ_h^U (i.e., 84 seconds) to γ_h (i.e., 72 seconds), the contribution of model h to the total revenue increases with a slope of PR_h. While the cycle time decreases from γ_h to γ_h^L (i.e., 64 seconds), the revenue increases with a slope of PR_h^{ϵ}. The slope between γ_h and γ_h^L is lower than the slope between γ_h^U and γ_h because after the production amount of the expected demand θ_h, each excess component of model h produced contributes an expected profit of PR_h^{ϵ}, which is lower than PR_h. Finally, when the cycle time decreases down from γ_h^L, the revenue contribution of model h remains constant because after the production amount of θ_h^L, any excess component of model h does not contribute any profit.

There are several reasons for an assembly line to stop when it is supposed to be operating. For instance, the tips of the welding tools may cling on the part during a welding operation and the welding tool then must be replaced with a new one. Also, the grippers used for transportation of parts through the assembly line may not close properly and therefore may not hold the part, which causes the assembly line to stop. Moreover, the censors may not recognize or may miscognize a part. In addition to these, a welding tool may finish its lifetime at some time when the assembly line is supposed to be operating. Except in the last instance, the reasons that cause the assembly line to stop are more technical and may happen at any time. However, the last instance can be prevented by making it possible to allow tool changes only in scheduled breaks such as tea or lunch breaks in which the assembly line does not operate.

In most of the automotive industries (including the one that we have been collaborating), plants work 24 hours in three 8-hour shifts and there are breaks in an 8-hour shift. In general, there is a 10-minute break at the beginning of a shift, a 10-minute tea break, a 30-minute lunch break and a second 10-minute tea break. The working hours between two breaks are equal and last 1 hour and 45 minutes. Each break corresponds to a possible tool change time period. In any break, if the remaining number of spots such that a welding tool can perform is less than the total number of spots it has to perform until the next break, it needs to be replaced with a new tool in that particular tool change time period in order to prevent line stoppages due to the tool changes. Let C_j be the cost of the tool in station j, $j \in M$. We define D_q as the available tool change time in tool change period q, $q = 1, 2, \ldots, U$, where U is the total number of breaks in the planning horizon.

We calculate the required tool change time as a function of the number of tools to be replaced:

$$K + \mu \cdot \sum_{j=1}^{m} z_{jq} \qquad \forall q = 1, 2, \ldots, U,$$

where K is a constant that denotes the time to prepare for welding tool changes and μ is another constant that corresponds to the time required to replace a single welding tool. Furthermore, z_{jq} is a binary variable to indicate whether tool in station $j \in M$ is changed in tool change period q. Two spot welding tools are attached to the spot welding gun as shown in Figure 2.3, and both of them are replaced at the same time.

As we have previously seen in Figure 2.1, the positions of spot welds on a part are at different locations. It may not be possible for a particular type of welding tool to reach to every location on a part due to its geometry. Therefore, it is not possible to process all spot welds by one type of welding tool so there

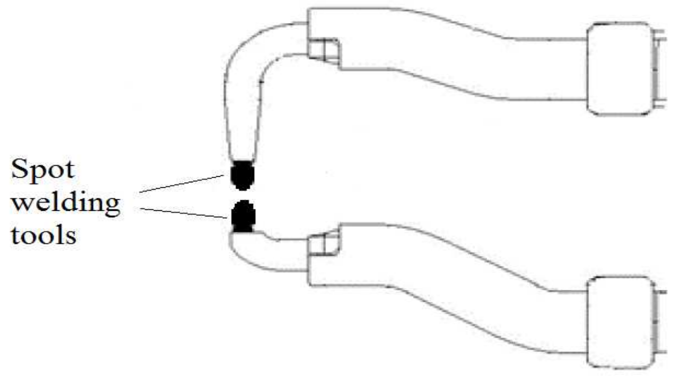


Figure 2.3: A spot welding gun

are several types of welding tools. As a result, a welding tool cannot perform all operations so we define an assignability matrix A and use the following parameter:

$$a_{hij} = \begin{cases} 1, & \text{if operation } i \in N_h \text{ of model } h \in G \text{ can be assigned} \\ & \text{to station } j \in M; \\ 0, & \text{otherwise.} \end{cases}$$

There may be additional subcomponents to be assembled to a part of model h inside the robotic cell. The subcomponents may block the positions of some spot welds required for an operation, i.e., O_{hi}. Therefore, O_{hi} must be performed before this particular subcomponent is assembled to the part. For similar reasons, we define a precedence matrix P and use the following parameter:

$$p_{hik} = \begin{cases} 1, & \text{if operation } i \in N_h \text{ of model } h \in G \text{ precedes operation} \\ & k \in N_h \text{ of model } h \in G; \\ 0, & \text{otherwise.} \end{cases}$$

The difference between the processing time of a spot weld by two different types of welding tools is negligible so the processing time of an operation is independent of the station at which it is processed. Therefore, we let t_{hi} be the time required to perform operation $i \in N$ of model $h \in G$. We calculate t_{hi} as a function of the number of spot welds required to perform operation i of model h

as follows:

$$t_{hi} = \alpha + \beta \cdot W_{hi} \qquad \forall h \in G, i \in N_h,$$

where α is a constant that denotes the time required for the robot to reach to the position to perform operation i of model h and β is another constant that corresponds to the time required to process a single spot weld.

In order to calculate the time allocated for production of parts of a particular model, we need to calculate the proportion of time that should be allocated to the production of a particular model. Therefore we define the parameter ψ_h for $h = 1, \ldots, g$ as the following:

$$\psi_h = \frac{\theta_h \cdot \gamma_h}{\sum_{l=1}^{g} \theta_l \cdot \gamma_l} \qquad \forall h \in G.$$

As mentioned before, the production time between two breaks is 1 hour 45 minutes, which is equal to 6300 seconds. Then, the time allocated for the production of parts of model h between two breaks in the assembly line is calculated as $6300 \cdot \psi_h$ seconds.

The decision variables we will use to formulate our model are defined below:

f_h: actual cycle time for model h, $h \in G$.

$$\sigma_j = \begin{cases} 1, & \text{if station } j \in M \text{ is used in the assembly line;} \\ 0, & \text{otherwise.} \end{cases}$$

$$x_{hij} = \begin{cases} 1, & \text{if operation } i \in N_h \text{ of model } h \in G \text{ is assigned to station } j \in M; \\ 0, & \text{otherwise.} \end{cases}$$

$$z_{jq} = \begin{cases} 1, & \text{if tool in station } j \in M \text{ is changed in tool change period } q, \\ & q = 1, 2, \ldots, U; \\ 0, & \text{otherwise.} \end{cases}$$

R_{jq}: remaining number of spot welds such that the welding tool in station j can process after tool change period q, $j = 1, 2, \ldots, m$, $q = 1, 2, \ldots, U$.

Having defined the parameters and the decision variables of our problem, the nonlinear mixed-integer mathematical model we will use to solve our problem is the following:

Model 1 (NLMIP):

$$\text{Maximize} \quad \sum_{h=1}^{g} PR_h \cdot \min\{\frac{\theta_h \cdot \gamma_h}{f_h}, \theta_h\} \tag{2.1}$$

$$+ \sum_{h=1}^{g} PR_h^{\epsilon} \cdot \min\{\max\{\frac{\theta_h \cdot \gamma_h}{f_h} - \theta_h, 0\}, \frac{\theta_h \cdot \gamma_h}{\gamma_h^L} - \theta_h\}$$

$$- \rho \cdot \sum_{j=1}^{m}\sum_{q=1}^{U} C_j \cdot z_{jq} - \eta \cdot \sum_{j=1}^{m} V_j \cdot \sigma_j$$

Subject to

$$f_h \geq \tau_j \cdot \sigma_j + \sum_{i=1}^{n_h} t_{hi} \cdot x_{hij} \qquad \forall h, j \tag{2.2}$$

$$f_h \leq \gamma_h^U \qquad \forall h \tag{2.3}$$

$$x_{hij} \leq a_{hij} \cdot \sigma_j \qquad \forall h, i, j \tag{2.4}$$

$$\sum_{j=1}^{m} x_{hij} = 1 \qquad \forall h, i \tag{2.5}$$

$$p_{hik} \cdot \sum_{l=1}^{j} x_{hil} + (1 - p_{hik}) \geq x_{hkj} \qquad \forall h, i, j, k \tag{2.6}$$

$$R_{jq} = B_j \cdot z_{jq} + [R_{j(q-1)} - \sum_{h=1}^{g} \frac{6300 \cdot \psi_h}{f_h} \cdot \sum_{i=1}^{n_h} W_{hi} \cdot x_{hij}](1 - z_{jq}) \qquad \forall j, q \tag{2.7}$$

$$R_{j0} = B_j \qquad \forall j \tag{2.8}$$

$$R_{jq} \geq \sum_{h=1}^{g} \frac{6300 \cdot \psi_h}{f_h} \cdot (\sum_{i=1}^{n_h} W_{hi} \cdot x_{hij}) \qquad \forall j, q \tag{2.9}$$

$$f_h \geq 0,\ R_{jq} \geq 0, \quad x_{hij}, \sigma_j, z_{jq} \in \{0, 1\} \qquad \forall h, i, j, q \tag{2.10}$$

We use a piecewise linear objective function in which we include the profit that is supposed to be earned by producing up to θ_h amount of parts of model h. We add the extra profit for the situation of producing more than θ_h and subtract the tooling cost incurred by replacing the welding tools and the cost incurred by setting up required stations. Tooling cost and station cost are converted to yearly costs by the constants ρ and η, which depend on the selection of U and the number of months that the production of the selected models will continue, respectively. Constraint (2.2) ensures that the cycle time is the maximum station time, which is the sum of the operation times allocated to that station plus a constant τ_j required for the robot in station j to begin and finalize processing the allocated operations. By (2.3), we prevent the actual cycle time for model h from exceeding γ_h^U. Constraint (2.4) restricts operations to be assigned only to the stations where they can be performed and to stations which are used in the assembly line. By (2.5), we ensure that an operation is assigned to exactly one station and none of the operations remains unassigned. Constraint (2.6) allows an operation to be assigned to a station if all its predecessor operations are assigned to the same or to a preceding station. Constraint (2.7) handles updates for the remaining number of spot welds for welding tools in stations through the production time. In Constraint (2.8), we define the remaining tool life for each tool at the beginning of the planning period. In this study, we assume that we always start the production with new tools. By constraint (2.9), we guarantee that none of the welding tools finishes its lifetime between two breaks since the cost of stopping the assembly line except the scheduled breaks is very costly.

In our problem, we assume that we have enough workforce to handle any number of tool changes in a particular tool change time period. In some cases, there may be restricted workforce to handle tool changes. Then, the following constraint, which ensures that we do not spend more than available time for tool changes in a particular tool change time period, can be added to Model 1:

$$K + \mu \cdot \sum_{j=1}^{m} z_{jq} \leq D_q \qquad \forall q = 1, 2, \ldots, U.$$

In this chapter, we have defined our problem, the problem parameters and the decision variables and formulated our problem as an NLMIP. In order to solve the NLMIP formulation of our problem, we use DICOPT, a nonlinear solver included in GAMS software. To work on more realistic instances, we provide data sets that are close to the data set which we worked on in the project that we had formerly conducted. However, as we will see in Chapter 6 in more detail, DICOPT fails to find solutions for the NLMIP instances with the given data sets. Therefore, in the next chapter of our study, we will convert NLMIP to a mixed integer problem by doing necessary linearizations for the objective function and nonlinear constraints and try to solve the equivalent mixed integer problem by using the commercial CPLEX solver.

Chapter 3

Linearization of NLMIP

NLMIP is a nonlinear mixed integer model that includes products of two or more variables. We continue our study with linearizing the objective function (e.g., equation (2.1)), constraint (2.2), constraint (2.3), constraint (2.7) and constraint (2.9) of NLMIP with some well-known linearization techniques.

First, we define the variable ω_h and replace it with $\frac{1}{f_h}$, which frequently occurs in Model 1. Now the nonlinear part

$$\sum_{h=1}^{g} PR_h \cdot \min\{\frac{\theta_h \cdot \gamma_h}{f_h}, \theta_h\} + \sum_{h=1}^{g} PR_h^{\epsilon} \cdot \min\{\max\{\frac{\theta_h \cdot \gamma_h}{f_h} - \theta_h, 0\}, \frac{\theta_h \cdot \gamma_h}{\gamma_h^L} - \theta_h\}$$

in our piecewise linear objective function becomes

$$\sum_{h=1}^{g} PR_h \cdot \min\{\theta_h \cdot \gamma_h \cdot \omega_h, \theta_h\}$$

$$+\sum_{h=1}^{g} PR_h^{\epsilon} \cdot \min\{\max\{\theta_h \cdot \gamma_h \cdot \omega_h - \theta_h, 0\}, \frac{\theta_h \cdot \gamma_h}{\gamma_h^L} - \theta_h\}. \qquad (1^*)$$

Introducing new positive variables AO_h, λ_h^1, λ_h^2, BO_h and a binary variable y_h, we propose Constraints (3.1)-(3.8) for the linearization of our objective function.

$$AO_h \leq \theta_h \cdot \gamma_h \cdot \omega_h \qquad \forall h \in G \qquad (3.1)$$

$$AO_h \leq \theta_h \qquad \forall h \in G \qquad (3.2)$$

$$\theta_h \cdot \gamma_h \cdot \omega_h - \theta_h = \lambda_h^1 - \lambda_h^2 \qquad \forall h \in G \qquad (3.3)$$

$$\lambda_h^1 \leq M \cdot y_h \qquad \forall h \in G \qquad (3.4)$$

$$\lambda_h^2 \leq M \cdot (1 - y_h) \qquad \forall h \in G \qquad (3.5)$$

$$BO_h \leq \lambda_h^1 \qquad \forall h \in G \qquad (3.6)$$

$$BO_h \leq \frac{\theta_h \cdot \gamma_h}{\gamma_h^L} - \theta_h \qquad \forall h \in G \qquad (3.7)$$

$$\omega_h, AO_h, BO_h, \lambda_h^1, \lambda_h^2 \geq 0,\ y_h \in \{0,1\} \qquad \forall h \in G \qquad (3.8)$$

Proposition 3.1. Constraints (3.1)-(3.8) correctly linearize the objective function introduced in NLMIP in Chapter 2, Equation (2.1).

Proof: First we replace $\min\{\theta_h \cdot \gamma_h \cdot \omega_h, \theta_h\}$ with AO_h. Since we deal with a maximization problem, Constraints (3.1) and (3.2) are sufficient for the linearization of the first term in (1*). Then, we replace $\max\{\theta_h \cdot \gamma_h \cdot \omega_h - \theta_h, 0\}$ that appears in the second term of (1*) with λ_h^1. By Constraints (3.3)-(3.5), we ensure that λ_h^1 is positive if $\theta_h \cdot \gamma_h \cdot \omega_h - \theta_h$ is positive, and 0 otherwise. Finally, we replace $\min\{\lambda_h^1, \frac{\theta_h \cdot \gamma_h}{\gamma_h^L} - \theta_h\}$ that appears in the second term of (1*) with BO_h. Similar to what we did for the linearization of the first term of (1*), Constraints (3.6) and (3.7) are sufficient for the linearization of the second term of (1*). Hence, Constraints (3.1)-(3.8) correctly linearize the objective function introduced in NLMIP in Equation (2.1). □

The nonlinear part of our objective function now becomes

$$\sum_{h=1}^{g} PR_h \cdot AO_h + \sum_{h=1}^{g} PR_h^{\epsilon} \cdot BO_h.$$

The new objective function, which is now linear, is the following:

$$\sum_{h=1}^{g} PR_h \cdot AO_h + \sum_{h=1}^{g} PR_h^{\epsilon} \cdot BO_h - \rho \cdot \sum_{j=1}^{m} \sum_{q=1}^{U} C_j \cdot z_{jq} - \eta \cdot \sum_{j=1}^{m} V_j \cdot \sigma_j$$

Using $\omega_h = \frac{1}{f_h}$, constraint (2.2) becomes

$$1 \geq \tau_j \cdot \omega_h + \sum_{i=1}^{n_h} t_{hi} \cdot x_{hij} \cdot \omega_h, \qquad \forall h, j. \tag{2.2*}$$

We define $e_{hij} = x_{hij} \cdot \omega_h$, a very large number F and propose Constraints (3.9)-(3.13) for the linearization of Constraint (2.2*).

$$e_{hij} \leq \omega_h \qquad \forall h \in G, i \in N_h, j \in M \tag{3.9}$$

$$e_{hij} \leq F \cdot x_{hij} \qquad \forall h \in G, i \in N_h, j \in M \tag{3.10}$$

$$e_{hij} \geq \omega_h - F \cdot (1 - x_{hij}) \qquad \forall h \in G, i \in N_h, j \in M \tag{3.11}$$

$$e_{hij} \geq 0 \qquad \forall h \in G, i \in N_h, j \in M \tag{3.12}$$

$$1 \geq \tau_j \cdot \omega_h + \sum_{i=1}^{n_h} t_{hi} \cdot e_{hij} \qquad \forall h \in G, j \in M \tag{3.13}$$

Proposition 3.2. Constraints (3.9)-(3.13) correctly linearize Constraint (2.2*).

Proof: First we replace $x_{hij} \cdot \omega_h$ with e_{hij}. We also know that ω_h is strictly positive for each $h \in G$ because cycle time of each model is strictly positive. So, if x_{hij} is equal to 1, by Constraints (3.9) and (3.11), $e_{hij} = \omega_h$. If x_{hij} is equal to 0, then by Constraints (3.10) and (3.12), e_{hij} is equal to 0. So in any case, e_{hij} is equal to $x_{hij} \cdot \omega_h$. Hence, Constraints (3.9)-(3.13) correctly linearize Constraint (2.2*). □

Since $\omega_h = \frac{1}{f_h}$, constraint (2.3) becomes

$$\frac{1}{\omega_h} \leq \gamma_h^U, \qquad \forall h \in G. \qquad (2.3^*)$$

We linearize constraint (2.3*) by replacing it with the following:

$$\gamma_h^U \cdot \omega_h \geq 1 \qquad \forall h. \tag{3.14}$$

As we previously defined $\omega_h = \frac{1}{f_h}$ and $e_{hij} = x_{hij} \cdot \omega_h$, constraint (2.7) becomes

$$R_{jq} = B_j \cdot z_{jq} + R_{j(q-1)} - R_{j(q-1)} \cdot z_{jq} - \sum_{h=1}^{g} 6300 \cdot \psi_h \cdot (\sum_{i=1}^{n_h} W_{hi} \cdot e_{hij})$$

$$+ \sum_{h=1}^{g} 6300 \cdot \psi_h \cdot (\sum_{i=1}^{n_h} W_{hi} \cdot e_{hij} \cdot z_{jq}) \qquad \forall j, q. \qquad (2.7^*)$$

Now we define $O_{jq} = R_{j(q-1)} \cdot z_{jq}$ and $\pi_{hijq} = e_{hij} \cdot z_{jq}$, and propose Constraints (3.15)-(3.23) for the linearization of Constraint (2.7*).

$$O_{jq} \leq R_{j(q-1)} \qquad \forall j \in M, q = 1, 2, \ldots, U \tag{3.15}$$

$$O_{jq} \leq F \cdot z_{jq} \qquad \forall j \in M, q = 1, 2, \ldots, U \tag{3.16}$$

$$O_{jq} \geq R_{j(q-1)} - F \cdot (1 - z_{jq}) \qquad \forall j \in M, q = 1, 2, \ldots, U \tag{3.17}$$

$$O_{jq} \geq 0 \qquad \forall j \in M, q = 1, 2, \ldots, U \tag{3.18}$$

$$\pi_{hijq} \leq e_{hij} \qquad \forall i \in N_h, j \in M, q = 1, 2, \ldots, U \tag{3.19}$$

$$\pi_{hijq} \leq F \cdot z_{jq} \qquad \forall i \in N_h, j \in M, q = 1, 2, \ldots, U \tag{3.20}$$

$$\pi_{hijq} \geq e_{hij} - F \cdot (1 - z_{jq}) \qquad \forall i \in N_h, j \in M, q = 1, 2, \ldots, U \tag{3.21}$$

$$\pi_{hijq} \geq 0 \qquad \forall i \in N_h, j \in M, q = 1, 2, \ldots, U \tag{3.22}$$

$$R_{jq} = B_j \cdot z_{jq} + R_{j(q-1)} - O_{jq} - \sum_{h=1}^{g} 6300 \cdot \psi_h \cdot (\sum_{i=1}^{n_h} W_{hi} \cdot e_{hij}) + \sum_{h=1}^{g} 6300 \cdot \psi_h \cdot (\sum_{i=1}^{n_h} W_{hi} \cdot \pi_{hijq}) \qquad \forall j \in M, q = 1, 2, \ldots, U \tag{3.23}$$

Proposition 3.3. Constraints (3.15)-(3.23) correctly linearize Constraint (2.7*).

Proof: First, we replace $R_{j(q-1)} \cdot z_{jq}$ with O_{jq}. Similar to the linearization of constraint (2.2*), by constraints (3.15)-(3.18), the third term of the right-hand side of Constraint (2.7*), $R_{j(q-1)} \cdot z_{jq}$, becomes linear. Then, we replace $e_{hij} \cdot z_{jq}$ that appears in the fifth term of the right-hand side of Constraint (2.7*) with π_{hijq}. Similar to the linearization of Constraint (2.2*) again, by Constraints (3.19)-(3.22), $e_{hij} \cdot z_{jq}$ becomes linear. Hence, Constraints (3.15)-(3.23) correctly linearize Constraint (2.7*). □

Using $e_{hij} = x_{hij} \cdot \omega_h$ again, constraint (2.9) becomes linear:

$$R_{jq} \geq \sum_{h=1}^{g} 6300 \cdot \psi_h \cdot (\sum_{i=1}^{n_h} W_{hi} \cdot e_{hij}) \qquad \forall j \in M, q = 1, 2, \ldots, U. \tag{3.24}$$

Having linearized the objective function and the necessary constraints, we finally have the following mixed integer model:

Model 2 (MIP):

$$
\begin{array}{ll}
\text{Maximize} & \displaystyle\sum_{h=1}^{g} PR_h \cdot AO_h + \sum_{h=1}^{g} PR_h^{\epsilon} \cdot BO_h \\
& \displaystyle - \rho \cdot \sum_{j=1}^{m}\sum_{q=1}^{U} C_j \cdot z_{jq} - \eta \cdot \sum_{j=1}^{m} V_j \cdot \sigma_j \\
\text{Subject to} & \text{Constraints (3.1)-(3.7)} \\
& \text{Constraints (3.9)-(3.11), (3.13)} \\
& \text{Constraint (3.14)} \\
& \text{Constraints (2.4)-(2.6)} \\
& \text{Constraints (3.15)-(3.17), (3.19)-(3.21), (3.23)} \\
& \text{Constraint (2.8)} \\
& \text{Constraint (3.24)} \\
& \omega_h, AO_h, BO_h, \lambda_h^1, \lambda_h^2 \geq 0,\ y_h \in \{0,1\}\ \ \forall h,\ \sigma_j \in \{0,1\}\ \ \forall j \\
& R_{jq}, O_{jq} \geq 0,\ z_{jq} \in \{0,1\}\ \ \forall j,q \\
& e_{hij} \geq 0,\ x_{hij} \in \{0,1\}\ \ \forall h, i \in N_h, j,\ \pi_{hijq} \geq 0\ \ \forall h, i \in N_h, j, q
\end{array}
$$

In summary, in this chapter, we have linearized our NLMIP model and obtained an MIP model by adding necessary linearization variables and constraints. As we will see further in Chapter 6, when we try to solve MIP by using CPLEX, we see that for several instances of MIP, CPLEX fails to find even a feasible solution to our problem in a time limit of 2 hours. Therefore, in the next chapter of our study, we will present a heuristic that finds a feasible solution for a given γ_h^U to instances of our problem. Later on, we will introduce an improvement procedure to obtain stronger results from the given initial feasible solutions.

Chapter 4

Proposed Heuristic Algorithm

As we have mentioned before, DICOPT and CPLEX fail to solve NLMIP and MIP instances, respectively and more detailed results for DICOPT and CPLEX will be given in Chapter 6. Therefore, in this chapter of our study, we will first present a heuristic algorithm that finds a set of initial feasible solutions for an instance of our problem. In the latter step, by adding a surrogate problem to our heuristic, we will improve the initial solutions in terms of the Total Profit value.

The heuristic algorithm that we will use for finding initial feasible solutions is called Algorithm 1. Algorithm 1 performs several major iterations, which result in distinct feasible solutions. Each major iteration starts with different cycle time upper bound (i.e., γ_h^U) values. Therefore, when we run Algorithm 1, by performing several major iterations, we perform a search for different initial solutions corresponding to different cycle time values over the intervals $[\gamma_h^L, \gamma_h^U]$ for $h \in G$. At each step of a major iteration, one operation is allocated to a station and at each step, the allocation performed satisfies the assignability, cycle time, precedence and tool life constraints.

The heuristic algorithm that we will use for improving the initial solutions is called Algorithm 2. At the end of each major iteration of Algorithm 2, we solve an additional mixed integer problem to improve the initial solution in terms of the total profit value. The additional mixed integer problem that we solve in

Algorithm 2 is a reduced version of the MIP given in Chapter 3, in which we replace our objective function with a surrogate one and use only the constraints that are necessary for ensuring the feasibility of an allocation found.

By obtaining a set of feasible solutions rather than a single solution, we gain more insight about the nonlinearity of our problem and the difficulty in solving it. Example 1 that will provide us more clear information about these points is solved by Algorithm 1, which we will soon present in Section 4.1.

Example 1: An automotive company is planning to set up a robotic cell assembly line to produce body components for two different models of cars ($g = 2$). Next year, the company plans to sell 150000 cars of the first model ($\theta_1 = 150000$) and 75000 cars of the second model ($\theta_2 = 75000$). To achieve this production amount, the company sets a target cycle time of 76 seconds for both models ($\gamma_h = 76$ for $h = 1, 2$). In order to meet the orders received so far, a cycle time of at most 80 seconds should be satisfied for both models ($\gamma_h^U = 80$ for $h = 1, 2$) and market research shows that the company can not sell cars more than the amount that can be produced when the cycle time is 60 seconds for both models ($\gamma_h^L = 60$ for $h = 1, 2$). Cost analysis shows that up to the production amount of 150000 for the first model and 75000 for the second model, each body component produced will contribute a profit of \$10 ($PR_h = 10$ for $h = 1, 2$). In case of producing more than the expected demand, each excess body component produced will contribute an expected profit of \$7 for both models ($PR_h^\epsilon = 7$ for $h = 1, 2$). In the production area, there is available space for at most ten stations ($m = 10$). The cost of setting up one station in the assembly line is \$192500 ($V_j = 192500$ for $j = 1, 2, \ldots, 10$). The welding tools have a tool life of 3200 spot welds ($B_j = 3200$ for $j = 1, 2, \ldots, 10$) and each welding tool costs \$10 ($C_j = 10$ for $j = 1, 2, \ldots, 10$). Both models require 15 spot welding operations ($n_h = 15$ for $h = 1, 2$) and the number of spots required to perform these operations are 10, 8, 8, 7, 10, 9, 8, 7, 10, 9, 7, 6, 13, 12 and 10, respectively. The time required for a robot in a station to reach to the position to perform an operation, the time required to perform a single spot weld and the time required for a robot to begin and finalize processing the allocated operations are calculated to be 1, 2 and 3 seconds, respectively ($\alpha = 1$, $\beta = 2$, $\tau_j = 3$ for $j = 1, 2, \ldots, 10$). An operation can be allocated to any station ($a_{hij} = 1$ for $h = 1, 2$, $i = 1, 2, \ldots, n_h$, $j = 1, 2, \ldots, 10$) and precedence relations between the welding operations are given in the following table.

Table 4.1: Precedence matrix for Example 1

h	$i \backslash k$	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	1	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0
1	2	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0
1	3	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0
1	4	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0
1	5	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0
1	6	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0
1	7	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0
1	8	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0
1	9	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1
1	10	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1
1	11	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1
1	12	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1
1	13	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	14	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	1	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0
2	2	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0
2	3	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0
2	4	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0
2	5	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0
2	6	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0
2	7	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0
2	8	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0
2	9	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1
2	10	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1
2	11	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1
2	12	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1
2	13	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	14	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

When we solve Example 1 with Algorithm 1, we get different solutions and objective function values corresponding to different cycle times. Figure 4.1 shows the values for the objective function terms and the objective function value as Total Profit for the corresponding cycle times.

As seen in Figure 4.1, revenue, which is the sum of the profits that each product contributes, strictly increases as the cycle times decrease until to γ_h^L

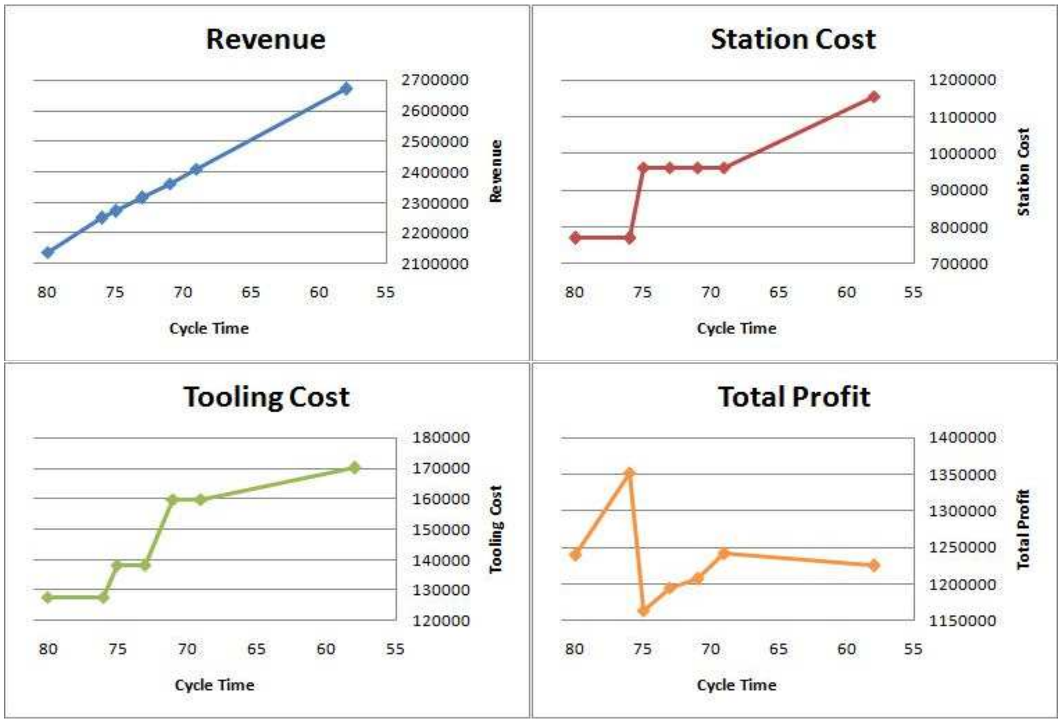


Figure 4.1: Cost and Profit Values for Example 1

values. Station cost is a nondecreasing function as long as the cycle times decrease because lower cycle times require a larger number of stations and it exhibits a stepping structure due to the breakpoints that correspond to particular cycle times.

After giving an insight about the solution of our problem and the nonlinearity of our objective function, we now present our heuristic algorithm that we use for finding initial solutions.

4.1 Heuristic For Finding Feasible Solutions

The heuristic that we introduce in this section consists of several major iterations. The only difference between these iterations are the values of parameter γ_h^U, the upper bound on the cycle time for model h, $h \in G$. Using different γ_h^U values at each major iteration result in a distinct feasible solution. In each major iteration, we start with creating a list I of unassigned operations, which initially includes all operations and becomes empty when all of the operations are allocated to the stations. Then, by using predecessor matrix P, for each operation O_{hi} we create $\text{pred}(h, i)$, a list that includes all predecessor operations of O_{hi}. In the further steps of each iteration, while allocating an operation to a station, we need to use the original $\text{pred}(h, i)$ lists to ensure that the allocation satisfies precedence constraints, but we also need to update these lists whenever an operation is allocated to a station. Therefore, for each O_{hi}, we create another predecessor list $\widehat{\text{pred}}(h, i)$, which is initially a copy of $\text{pred}(h, i)$ but is subject to changes in the further steps. The reason for keeping dynamic $\widehat{\text{pred}}(h, i)$ lists is that an operation becomes assignable when its $\widehat{\text{pred}}(h, i)$ list is empty. While assigning the operations to the stations, for each station, we need to keep information about the total time spent and total number of spots performed for a particular model in order to ensure that cycle time and tool life constraints remain satisfied. Therefore, we use $load_{hj}$ and $spot_{hj}$, which keep the necessary information about the total time spent and total number of spots performed for a particular model in each station and which are initially equal to τ_j and 0, respectively.

After these initializations, we start allocating the welding operations to stations. At each step, we allocate only one operation to a station. Therefore, we perform $\sum_{h=1}^{g} n_h$ steps until there is no operation left to allocate, in other words, $I = \emptyset$. At each step, we choose to allocate the operation O_{hi} with the largest operation time and whose predecessor operations have all been allocated before. Our selection rule resembles using Longest Processing Time (LPT) rule, which is one of the most popular techniques used in bin packing problems. By allocating the operation with the largest processing time, we aim to have a solution with as few stations as possible, in other words, with as low investment costs as possible. We allocate such a candidate operation to the station with minimum possible index, while ensuring the feasibility of four important constraints. Firstly, the candidate operation should be assignable to that station. Secondly, after allocating the candidate operation, the cycle time for that particular model should not be exceeded. Thirdly, we check if each predecessor operation of the candidate operation has been assigned to the same or to a preceding station. Finally, we ensure that after allocating the candidate operation, the total number of spots that will be performed at that station between two breaks will not exceed the life of the welding tool at the same station. Once we allocate an operation to a station, we remove it from I, the list of unassigned operations. We update $load_{hj}$ and $spot_{hj}$ for the station that the operation is allocated and remove the operation from the predecessor lists $\widehat{\text{pred}}(h, i)$ of those other operations for which the allocated operation is a predecessor operation. When the allocation of all operations is complete, we calculate objective function value for the corresponding solution.

As mentioned before, our heuristic consists of several major iterations. We start the first major iteration with $\widehat{\gamma}_h^U = \gamma_h^U$ values for each $h \in G$. At the end of each major iteration, we obtain cycle time (f_h) values and start the next major iteration with $\widehat{\gamma}_h^U = f_h - 1$. This setting provides us to obtain different solutions from each major iteration and we continue this process until $\widehat{\gamma}_h^U < \gamma_h^L$ for each $h \in G$. The reason behind this stopping criterion is that achieving a cycle time f_h lower than γ_h^L for any model does not contribute any additional profit and therefore the objective function value does not increase.

In Example 1 that was presented at the beginning of this chapter, the largest objective function value corresponds to the solution in which $f_h = \gamma_h$ for both models. But since all objective function terms depend on the values of cycle times, this may not always be the case. In other words, we may obtain larger objective function values with arbitrary cycle times. The following example shows the importance of why we perform a search for different solutions corresponding to different cycle time values over the intervals $[\gamma_h^L, \gamma_h^U]$ for $h \in G$.

Example 2: Suppose that the company in Example 1 can not sell cars more than the amount that can be produced when the cycle time is 65 seconds instead of 60 seconds for both models ($\gamma_h^L = 65$ for $h = 1, 2$). Furthermore, the cost of setting up one station in the assembly line is decreased to \$110000 ($V_j = 110000$ for $j = 1, 2, \ldots, 10$) instead of \$192500 in the previous example. All the data provided in Example 1 except these two remain the same. For Example 2, Figure 4.2 shows the values for the objective function terms and the objective function

Algorithm 1: Heuristic for finding initial feasible solutions

```
Input: $m \in \mathbb{N}$, $g \in \mathbb{N}$, $n_h$ for $h = 1, \ldots, g$, A, P, $W_{hi}$ for $h = 1, \ldots, g$,
       $i = 1, \ldots, n_h$, $B_j$ and $\tau_j$ for $j = 1, \ldots, m$, $\gamma_h$, $\gamma_h^U$ and $\theta_h$ for $h = 1, \ldots, g$.
Output: $S$, a set of feasible solution(s)
begin
    initialize
        $\widehat{\gamma}_h^U = \gamma_h^U \ \ \forall h \in G$
        $S = \emptyset$
    while $\exists\, \widehat{\gamma}_h^U$ such that $\widehat{\gamma}_h^U \geq \gamma_h^L$ do
        Create list $I$ that contains all operations $O_{hi}$, $h = 1, \ldots, g$, $i = 1, \ldots, n_h$
        Create predecessor list $\text{pred}(h, i)$ for each $O_{hi}$, $h \in G$, $i \in N_h$ from $P$
        initialize
            $\widehat{\text{pred}}(h, i) = \text{pred}(h, i)$, $load_{hj} = \tau_j, spot_{hj} = 0$ for all $h = 1, \ldots, g$,
            $i = 1, \ldots, n_h, j = 1, \ldots, m$
        while $I \neq \emptyset$ do
            Find an operation $O_{hi} \in I$ with $\widehat{\text{pred}}(h, i) = \emptyset$ and maximum $t_{hi}$
            Assign $O_{hi}$ to station $S_j$ with minimum possible index such that
                $a_{hij} = 1$
                $load_{hi} + t_{hi} \leq \gamma_h^U$
                $p_{hik} \cdot \sum_{l=1}^{j} x_{hil} + (1 - p_{hik}) \geq x_{hkj}$ for each $k \in \text{pred}(h, i)$
                $\frac{6300 \cdot \psi_h}{\max\{\max_{k \in M \setminus j}\{load_{hk}\}, load_{hj} + t_{hi}\}} \cdot (spot_{hj} + W_{hi})$
                  $+ \sum_{\bar{h} \in G \setminus h} \frac{6300 \cdot \psi_{\bar{h}}}{\max_{k \in M}\{load_{\bar{h}k}\}} \cdot spot_{\bar{h}j} \leq B_j$
            $load_{hj} = load_{hj} + t_{hi}$
            $spot_{hj} = spot_{hj} + W_{hi}$
            $I = I \setminus \{O_{hi}\}$
            $\widehat{\text{pred}}(c, d) = \widehat{\text{pred}}(c, d) \setminus \{O_{hi}\}$ for all $O_{cd} \in I$
        Let $y$ be the corresponding solution
        Find $\varphi_y$, the corresponding objective function value
        $S = S \cup y$
        $f_h \leftarrow \max_{j \in M}\{load_{hj}\} \ \ \forall h \in G$
        $\widehat{\gamma}_h^U \leftarrow f_h - 1 \ \ \forall h \in G$
end
```

value as Total Profit for the corresponding cycle times.

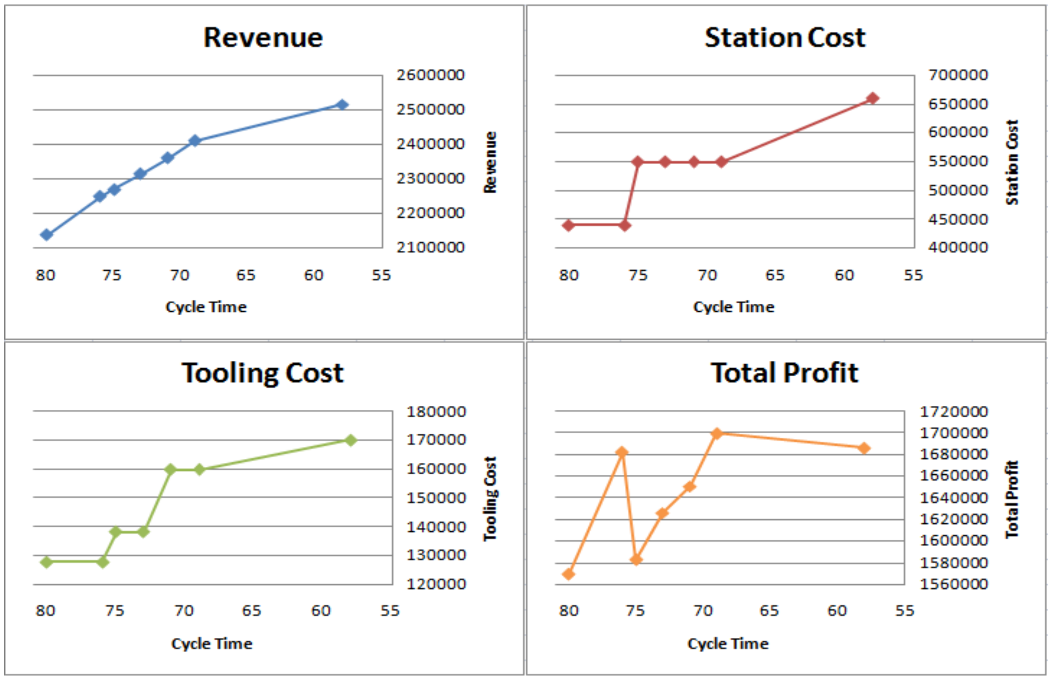


Figure 4.2: Cost and Profit Values for Example 2

The two different Total Profit value patterns in Figure 4.1 and Figure 4.2 show that our problem is difficult in the sense that Total Profit is a ‘very nonlinear function’ according to the DICOPT solutions manual. Also, Figure 4.2 clearly shows that there may be multiple peak points of Total Profit function, which means that the global optimum for the objective function value may correspond to particular cycle times in the intervals $[\gamma_h^L, \gamma_h^U]$ for $h \in G$ and even for the instances of NLMIP that can be solved by DICOPT, there is the fact that DICOPT may get stuck in one of the local optima. Therefore, we perform a search for alternative solutions corresponding to different cycle time values over the intervals $[\gamma_h^L, \gamma_h^U]$ for $h \in G$. In addition to these facts, we still do not know whether we find the best solution achievable in terms of the objective function value. Hence, in the following section of our study, we will try to strengthen our search methods in order to find better solutions than the ones that we can find with Algorithm 1.

4.2 Improvement Algorithm

Algorithm 1 presented in Section 4.1 does not necessarily give an optimal allocation of operations in terms of the objective function value of MIP. In other words, there may be a better allocation of operations that corresponds to cycle times different than the ones that correspond to a solution found by Algorithm 1.

As previously defined, let $y \in S$ be a feasible solution that is found by Algorithm 1 to our problem. Let m_y be the number of stations used in the assembly line in the feasible solution y. As long as the cycle times are larger than the corresponding γ_h^L values, achieving lower cycle times with less than or equal to m_y stations increases the revenue and so the Total Profit, which we aim to maximize. Hence, in this section, we present the following problem, in which we use a surrogate objective function to minimize the cycle time of each model to be produced in addition to the overall cycle time denoted by MaxCycleTime.

$$
\begin{array}{lll}
 & \text{Minimize} & \text{MaxCycleTime} + \sum_{h=1}^{g} f_h - \gamma_h^L \\
 & \text{Subject to} & \text{MaxCycleTime} \geq f_h \quad \forall h \\
 & & \text{Constraint (2.2)} \\
(\text{SP}) & & \sum_{j=1}^{m} \sigma_j \leq m_y \\
 & & \text{Constraints (3.9)-(3.11), (3.13), (3.14)} \\
 & & \text{Constraints (2.4)-(2.6)} \\
 & & B_j \geq \sum_{h=1}^{g} 6300 \cdot \psi_h \cdot (\sum_{i=1}^{n_h} W_{hi} \cdot e_{hij}) \quad \forall j \\
 & & \text{MaxCycleTime} \geq 0,\ f_h, \omega_h \geq 0\ \ \forall h, \\
 & & e_{hij} \geq 0,\ x_{hij} \in \{0,1\}\ \ \forall h, i \in N_h, j,\ \sigma_j \in \{0,1\}\ \ \forall j
\end{array}
$$

In order to achieve better allocations of operations in terms of the objective function value, we reduce our problem to SP. The optimal allocation to SP is still a feasible solution for MIP that is presented in Chapter 3 as shown in the following lemma.

Lemma: Let $\bar{x}$ be an optimal solution to SP, then $\bar{x}$ is a feasible solution to MIP.

Proof: In SP, we omit constraints (3.1)-(3.8), (3.15)-(3.17), (3.19)-(3.21), (3.23), (2.8) and (3.24), which are present in MIP. We need constraints (3.1)-(3.8) for linearization of the objective function of NLMIP. Since our objective in SP is different from MIP, omitting these constraints does not affect the feasibility of $\bar{x}$ to MIP. We use constraints (3.15)-(3.17), (3.19)-(3.21) and (3.23) in MIP for linearization of constraint (2.7) in NLMIP. Together with constraints (2.8) and (3.24), these constraints ensure that the remaining number of spot welds that a welding tool in station j can perform after tool change time period q is greater than or equal to the number of spot welds it has to perform until the next tool change time period. In other words, we need these constraints to determine the values of z_{jq} variables, which are necessary for calculation of tooling costs in the objective function. Again, since we have a different objective function and z_{jq} values are necessary for calculation of tooling costs, omitting constraints (3.15)-(3.17), (3.19)-(3.21), (3.23), (2.8) and (3.24) does not affect the feasibility of $\bar{x}$ to MIP. $\square$

Let $y \in S$ be a feasible solution that is found by Algorithm 1 to our problem and let m_y be the number of stations used in the assembly line in the feasible solution y. We first run SP for $m_y - 1$ stations with the same upper bound and other parameters used in Algorithm 1 that result in the feasible solution y. If the same cycle times in the feasible solution y can not be met by using less number of stations, we run SP again for m_y stations to improve the cycle times and so the Total Profit that we want to maximize in the original problem.

Let $y \in S$ be a feasible solution that is found by Algorithm 1 to our problem and let $\bar{y}$ be an optimal SP solution using the same parameters that result in the feasible solution y in Algorithm 1. Let φ_y and $\varphi_{\bar{y}}$ be the objective function values that correspond to y and $\bar{y}$, respectively. Algorithm 2 is a combination of Algorithm 1 and SP.

As mentioned in the previous chapter of our study, the difficulty of our problem results from the fact that our objective function is very nonlinear. In the following example, we will see how DICOPT fails to find the optimal solution and how we

Algorithm 2: Algorithm for improving heuristic solutions

```
Input: $m \in \mathbb{N}$, $g \in \mathbb{N}$, $n_h$ for $h = 1, \ldots, g$, A, P, $W_{hi}$ for $h = 1, \ldots, g$,
       $i = 1, \ldots, n_h$, $B_j$ and $\tau_j$ for $j = 1, \ldots, m$, $\gamma_h$, $\gamma_h^U$ and $\theta_h$ for $h = 1, \ldots, g$.
Output: Best solution
begin
    initialize
        $\widehat{\gamma}_h^U = \gamma_h^U \ \forall h \in G$
        $S = \emptyset, \bar{S} = \emptyset$
    while $\exists\ \widehat{\gamma}_h^U$ such that $\widehat{\gamma}_h^U \geq \gamma_h^L$ do
        Create list $I$ that contains all operations $O_{hi}$, $h = 1, \ldots, g$, $i = 1, \ldots, n_h$
        Create predecessor list pred$(h, i)$ for each $O_{hi}$, $h \in G$, $i \in N_h$ from $P$
        initialize
            $\widehat{\text{pred}}(h, i) = \text{pred}(h, i)$, $load_{hj} = \tau_j, spot_{hj} = 0$ for all $h = 1, \ldots, g$,
            $i = 1, \ldots, n_h, j = 1, \ldots, m$
        while $I \neq \emptyset$ do
            Find an operation $O_{hi} \in I$ with $\widehat{\text{pred}}(h, i) = \emptyset$ and maximum $t_{hi}$
            Assign $O_{hi}$ to station $S_j$ with minimum possible index such that
                $a_{hij} = 1$
                $load_{hi} + t_{hi} \leq \gamma_h^U$
                $p_{hik} \cdot \sum_{l=1}^{j} x_{hil} + (1 - p_{hik}) \geq x_{hkj}$ for each $k \in \text{pred}(h, i)$
                $\frac{6300 \cdot \psi_h}{\max\{\max_{k \in M \setminus j}\{load_{hk}\}, load_{hj} + t_{hi}\}} \cdot (spot_{hj} + W_{hi})$
                  $+ \sum_{\bar{h} \in G \setminus h} \frac{6300 \cdot \psi_{\bar{h}}}{\max_{k \in M}\{load_{\bar{h}k}\}} \cdot spot_{\bar{h}j} \leq B_j$
            $load_{hj} = load_{hj} + t_{hi}$
            $spot_{hj} = spot_{hj} + W_{hi}$
            $I = I \setminus \{O_{hi}\}$
            $\widehat{\text{pred}}(c, d) = \widehat{\text{pred}}(c, d) \setminus \{O_{hi}\}$ for all $O_{cd} \in I$
        Let $y$ be the corresponding feasible solution
        $S = S \cup y$
        Solve the SP model with the same parameters and find $\bar{y}$, the
        corresponding SP solution
        $\bar{S} = \bar{S} \cup \bar{y}$
        $f_h \leftarrow \max_{j \in M}\{load_{hj}\} \ \forall h \in G$
        $\widehat{\gamma}_h^U \leftarrow f_h - 1 \ \forall h \in G$
    Find $\varphi_y$ and $\varphi_{\bar{y}}$ for each $y \in S$ and $\bar{y} \in \bar{S}$
    $Best \leftarrow \text{argmax}_{y \in S, \bar{y} \in \bar{S}}\{\max\{\max_{y \in S}\varphi_y, \max_{\bar{y} \in \bar{S}}\varphi_{\bar{y}}\}\}$
end
```

improve our heuristic results by adding the surrogate problem.

Example 3: Suppose that the company in Example 1 has an available space in their production area only for four stations ($m = 4$). Suppose further that both models require 10 spot welding operations ($n_h = 10$ for $h = 1, 2$) and the number of spot welds required to perform these operations are 10, 8, 8, 7, 10, 9, 8, 7, 10 and 9, respectively. Except precedence relations, all the data provided in Example 1 remain the same. Precedence relations for operations are given in Table 4.2.

Table 4.2: Precedence matrix for Example 3

h	$i\backslash^k$	1	2	3	4	5	6	7	8	9	10
1	1	0	0	0	0	1	1	1	1	0	0
1	2	0	0	0	0	1	1	1	1	0	0
1	3	0	0	0	0	1	1	1	1	0	0
1	4	0	0	0	0	1	1	1	1	0	0
1	5	0	0	0	0	0	0	0	0	1	1
1	6	0	0	0	0	0	0	0	0	1	1
1	7	0	0	0	0	0	0	0	0	1	1
1	8	0	0	0	0	0	0	0	0	1	1
1	9	0	0	0	0	0	0	0	0	0	0
1	10	0	0	0	0	0	0	0	0	0	0
2	1	0	0	0	0	1	1	1	1	0	0
2	2	0	0	0	0	1	1	1	1	0	0
2	3	0	0	0	0	1	1	1	1	0	0
2	4	0	0	0	0	1	1	1	1	0	0
2	5	0	0	0	0	0	0	0	0	1	1
2	6	0	0	0	0	0	0	0	0	1	1
2	7	0	0	0	0	0	0	0	0	1	1
2	8	0	0	0	0	0	0	0	0	1	1
2	9	0	0	0	0	0	0	0	0	0	0
2	10	0	0	0	0	0	0	0	0	0	0

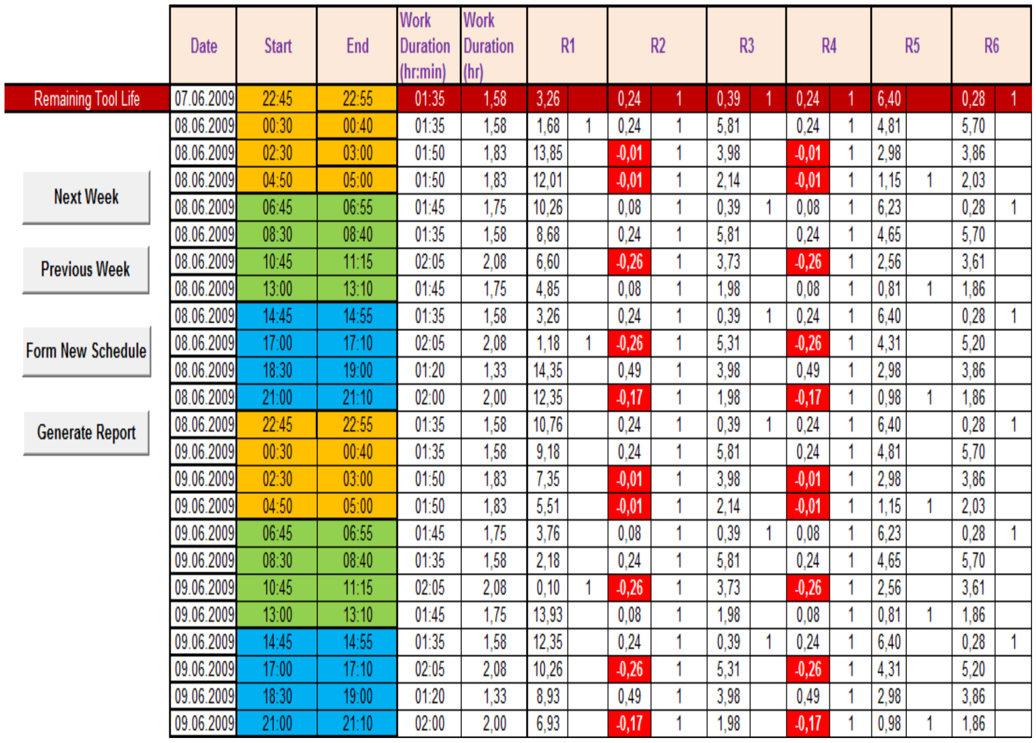
In this example, the best objective function value that is achievable by Algorithm 1 is 1640022 and in this solution, the number of stations used in the assembly line is equal to 3 and cycle times for both models are equal to 73. However, when we run SP, we see that with 3 stations, a cycle time of 69 is achievable for both models. Such a decrease in the cycle time results in more production and so in more revenue. Hence, the best objective value achievable increases to 1735082 by Algorithm 2. Furthermore, the best solution found for this instance by DICOPT has an objective function value of 1702947, which means that DICOPT got stuck in one of the local optima of the objective function. Moreover, this relatively small instance of our problem can be solved to optimality by CPLEX in approximately 20 minutes and the optimal value is the same with the one that we find by Algorithm 2. But for this instance, Algorithm 2 finds the optimal solution just in a few seconds.

In this chapter, we have given our proposed heuristic algorithm together with some examples and numerical results. In the next chapter, we will see an overview of the implementation of our tool change decisions to a real-life assembly line problem. In Chapter 6, we will test the performance of our heuristic on some instances of our problem and compare the performances of our heuristic and DICOPT and CPLEX solvers.

Chapter 5

Implementation

In this chapter, we will give some details about the tool change decision policy that we proposed to solve the real-life assembly line problem that we faced in the project that we had formerly conducted at one of the leading automotive companies in Turkey. As mentioned before, while defining our problem and constraints in our study, we were inspired by this project. Our aim in the project was to increase the efficiency of an assembly line that performed spot welding operations and produced body components for different models of cars. The most important problem of the assembly line was that some tool changes coincided to the times when the assembly line was supposed to be operating. Therefore, in order to perform tool changes, the assembly line was stopped and this resulted in loss of production. In their previous assembly line balancing problems, the company has allocated almost 10% of their available capacity to the line stoppages due to the tool changes. To eliminate such tool change related line stoppages, we developed a decision support system that indicated at which break each of the welding tools in the assembly line should be replaced with a new one. All the relevant information about operation allocations, number of spot welds that each tool must perform and cycle times for each model was provided to us by the company. However, since the assembly line was operating for a long time, we did not have a chance to revise the allocation of welding operations to the stations because it would take a long time to recode the robot operations and movements and to perform production simulation studies. Therefore, we were able to apply the results of our tool change analysis only to the current system, without changing any operation allocation. Figure 5.1 is a screenshot of one of the modules that we had in our decision support system.



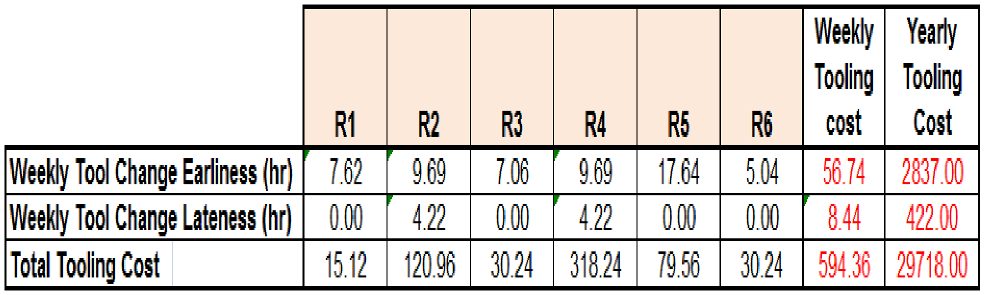
	Date	Start	End	Work Duration (hr:min)	Work Duration (hr)	R1		R2		R3		R4		R5		R6	
Remaining Tool Life	07.06.2009	22:45	22:55	01:35	1,58	3,26		0,24	1	0,39	1	0,24	1	6,40		0,28	1
	08.06.2009	00:30	00:40	01:35	1,58	1,68	1	0,24	1	5,81		0,24	1	4,81		5,70	
	08.06.2009	02:30	03:00	01:50	1,83	13,85		-0,01	1	3,98		-0,01	1	2,98		3,86	
	08.06.2009	04:50	05:00	01:50	1,83	12,01		-0,01	1	2,14		-0,01	1	1,15	1	2,03	
	08.06.2009	06:45	06:55	01:45	1,75	10,26		0,08	1	0,39	1	0,08	1	6,23		0,28	1
	08.06.2009	08:30	08:40	01:35	1,58	8,68		0,24	1	5,81		0,24	1	4,65		5,70	
	08.06.2009	10:45	11:15	02:05	2,08	6,60		-0,26	1	3,73		-0,26	1	2,56		3,61	
	08.06.2009	13:00	13:10	01:45	1,75	4,85		0,08	1	1,98		0,08	1	0,81	1	1,86	
	08.06.2009	14:45	14:55	01:35	1,58	3,26		0,24	1	0,39	1	0,24	1	6,40		0,28	1
	08.06.2009	17:00	17:10	02:05	2,08	1,18	1	-0,26	1	5,31		-0,26	1	4,31		5,20	
	08.06.2009	18:30	19:00	01:20	1,33	14,35		0,49	1	3,98		0,49	1	2,98		3,86	
	08.06.2009	21:00	21:10	02:00	2,00	12,35		-0,17	1	1,98		-0,17	1	0,98	1	1,86	
	08.06.2009	22:45	22:55	01:35	1,58	10,76		0,24	1	0,39	1	0,24	1	6,40		0,28	1
	09.06.2009	00:30	00:40	01:35	1,58	9,18		0,24	1	5,81		0,24	1	4,81		5,70	
	09.06.2009	02:30	03:00	01:50	1,83	7,35		-0,01	1	3,98		-0,01	1	2,98		3,86	
	09.06.2009	04:50	05:00	01:50	1,83	5,51		-0,01	1	2,14		-0,01	1	1,15	1	2,03	
	09.06.2009	06:45	06:55	01:45	1,75	3,76		0,08	1	0,39	1	0,08	1	6,23		0,28	1
	09.06.2009	08:30	08:40	01:35	1,58	2,18		0,24	1	5,81		0,24	1	4,65		5,70	
	09.06.2009	10:45	11:15	02:05	2,08	0,10	1	-0,26	1	3,73		-0,26	1	2,56		3,61	
	09.06.2009	13:00	13:10	01:45	1,75	13,93		0,08	1	1,98		0,08	1	0,81	1	1,86	
	09.06.2009	14:45	14:55	01:35	1,58	12,35		0,24	1	0,39	1	0,24	1	6,40		0,28	1
	09.06.2009	17:00	17:10	02:05	2,08	10,26		-0,26	1	5,31		-0,26	1	4,31		5,20	
	09.06.2009	18:30	19:00	01:20	1,33	8,93		0,49	1	3,98		0,49	1	2,98		3,86	
	09.06.2009	21:00	21:10	02:00	2,00	6,93		-0,17	1	1,98		-0,17	1	0,98	1	1,86	

Figure 5.1: Tool change schedules for spot welding tools

Figure 5.1 shows a part of a weekly tool change schedule for spot welding tools. The four buttons on the left are used for returning back to the tool change schedule of the previous week, advancing to the next production week and forming the tool change schedule and generating a tool change report. The dates and the starting and ending hours of each break are given. The work duration column shows the production time between the current break and the next break. R1, R2, R3, R4, R5 and R6 represent the spot welding tools at six different stations. Under these columns, the remaining life of welding tools after each break is given. If the number '1' occurs in any cell of these columns, it means that the welding tool has to be changed with a new one at the corresponding break because the remaining life of the tool is not enough for that tool to be used for production until the next break.

If the remaining life of a tool after a particular break is negative and hence the corresponding cell is highlighted (i.e., filled in red), it means that the life of such a tool has expired at some time before the break. Although the tool change decisions are made to eliminate such occurrences and hence tool change related line stoppages, it did not work for all the spot welding tools in the current system. This might occur for two reasons. In initial allocation of the operations, there might be some spot welding tools with a very frequent usage such that the tool life might be shorter than the available time period. Another reason is that our decision support system uses real time data and once it is executed, it obtains the current time and the remaining number of spot welds that each tool can perform from Welding Management System (WMS), version 2.94, which is a software developed for welding operations by Gf Welding S.p.A. company. Therefore, there might be some deviation between the actual number of components being produced and the expected demand information that we have used in our formulation due to the unexpected events leading to line stoppages such as the tips of the welding tools may cling on a part during a welding operation, the grippers used for transportation of parts through the assembly line may not close properly or the censors may not recognize or may miscognize a part. For example, although the welding tools R2 and R4 are changed at every break, they still sometimes cause the assembly line to stop between two particular breaks. The reason for such stoppages was that the number of spots that the welding tools R2 and R4 had to perform between two particular breaks as a result of the predetermined operation allocations sometimes exceeded their tool lives in terms of the total number of spot welds that they can perform. Therefore, it can be derived that tool change related stoppages in assembly lines cause from allocation of operations which are determined without considering the limited lives of tools that are used for production. Hence, this is the reason why we determine operation allocations and tool change decisions jointly in our study.

Another useful property of our decision support system is that one can see the corresponding tooling cost that is incurred for particular tool change schedules. Figure 5.2 shows the weekly and yearly tooling costs for a particular tool change schedule.

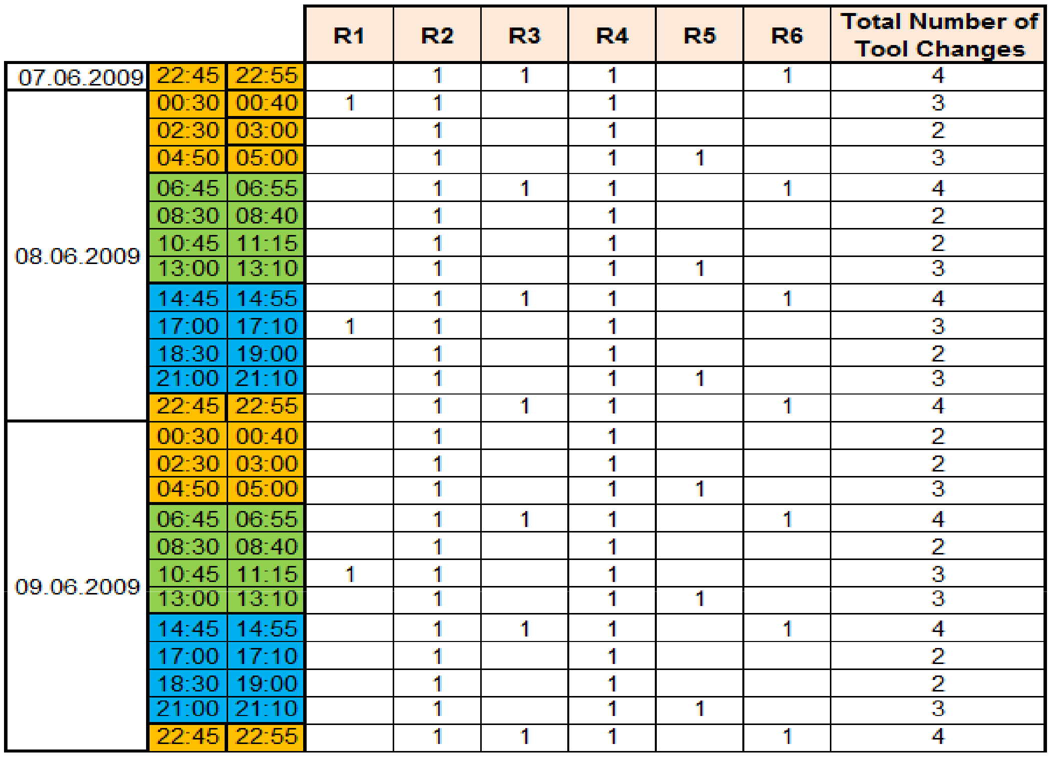


	R1	R2	R3	R4	R5	R6	Weekly Tooling cost	Yearly Tooling Cost
Weekly Tool Change Earliness (hr)	7.62	9.69	7.06	9.69	17.64	5.04	56.74	2837.00
Weekly Tool Change Lateness (hr)	0.00	4.22	0.00	4.22	0.00	0.00	8.44	422.00
Total Tooling Cost	15.12	120.96	30.24	318.24	79.56	30.24	594.36	29718.00

Figure 5.2: Tooling Cost Calculations

The tooling cost calculations enable the company to do sensitivity analysis for tool change decisions. As seen in Figure 5.2, in the current system, there are some tools that are changed before the end of their tool lives (i.e., earliness > 0) and some other tools that cause the assembly line to stop (i.e., lateness > 0). As we have observed in the actual production, if they could have reallocated one or two operations from station 4 to station 3 (which was feasible due to the cycle time, precedence and assignability constraints), both the earliness cost of station 3 and the lateness cost of station 4 could be decreased at the same time. Therefore, the current allocation could easily be improved in terms of the tooling cost leading to a higher total profit value since the other parameters in the objective function remain the same. If the tool change earliness is high and the reallocation is not possible due to either cycle time or precedence or assignability constraints, the company may set up automated tool changers in the assembly line. If this option is realized, there might be some loss in the total available production time due to decreasing the earliness because the tools that are changed early will be changed at some time when the assembly line is supposed to be operating, rather than in breaks. However, there might also be some gain in the total available production time due to decreasing the lateness because automated tool changers will change the tool in relatively shorter time with respect to stopping the assembly line and changing the tools manually. In addition to these, automated tool changers may decrease the reliability of the tool changes and hence the quality of the products. Moreover, investment costs will be incurred for setting up tool changers in the assembly lines. However, increasing the utilization of the spot welding tools would decrease the total tooling cost. Therefore, sensitivity analysis that can be handled via our decision support system is very useful for evaluating such options.

In addition to providing tool change schedules, we provided a summary of tool changes and the total number of tools to be changed at each break as a tool change support system. Figure 5.3 shows another module of our decision support system, which indicates necessary tool changes to be performed at each break.

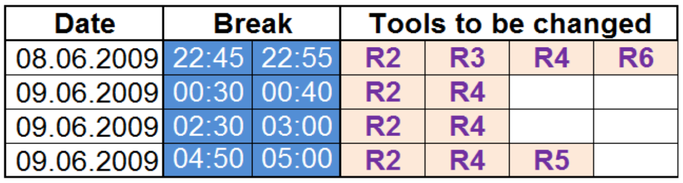


			R1	R2	R3	R4	R5	R6	Total Number of Tool Changes
07.06.2009	22:45	22:55		1	1	1		1	4
08.06.2009	00:30	00:40	1	1		1			3
	02:30	03:00		1		1			2
	04:50	05:00		1		1	1		3
	06:45	06:55		1	1	1		1	4
	08:30	08:40		1		1			2
	10:45	11:15		1		1			2
	13:00	13:10		1		1	1		3
	14:45	14:55		1	1	1		1	4
	17:00	17:10	1	1		1			3
	18:30	19:00		1		1			2
	21:00	21:10		1		1	1		3
	22:45	22:55		1	1	1		1	4
09.06.2009	00:30	00:40		1		1			2
	02:30	03:00		1		1			2
	04:50	05:00		1		1	1		3
	06:45	06:55		1	1	1		1	4
	08:30	08:40		1		1			2
	10:45	11:15	1	1		1			3
	13:00	13:10		1		1	1		3
	14:45	14:55		1	1	1		1	4
	17:00	17:10		1		1			2
	18:30	19:00		1		1			2
	21:00	21:10		1		1	1		3
	22:45	22:55		1	1	1		1	4

Figure 5.3: Summary of spot welding tool changes

Figure 5.3 shows only the tool change times for each spot welding tool and the total number of tools to be changed at each break. The values in the total number of tool changes column are also used for calculating the total weekly tooling cost.

In order to send the tool change decisions to the production area, we provided an interface to be monitored in the production area in order for the workers to obtain necessary information for tool change schedules. Figure 5.4 shows a shift-based summary of tool changes. Based on the current data obtained at any time, our module determines and reports the necessary welding tool changes for each break of the current shift. When monitored in the production area, this information also provides the workers an ease for tracing and performing tool changes.



Date	Break		Tools to be changed			
08.06.2009	22:45	22:55	R2	R3	R4	R6
09.06.2009	00:30	00:40	R2	R4		
09.06.2009	02:30	03:00	R2	R4		
09.06.2009	04:50	05:00	R2	R4	R5	

Figure 5.4: A monitoring of shift-based tool changes

Chapter 6

Computational Study

In this chapter of our study, we will evaluate the performances of our heuristic method and the commercial solvers DICOPT2X-c and CPLEX 10.1 for our problem and provide some computational results that we obtained by using Algorithm 1 and Algorithm 2 that are coded in C++ environment, and the solvers mentioned above. We used the modeling interface of GAMS 22.3. A computer with 1024 MB memory and Pentium Dualcore 1.73 GHZ CPU was used for taking the runs. First of all, we define the data set that we used in our runs.

There are two important factors that affect the size of our problem and optimal allocations in the solutions. The first one is the parameter g, the number of different models that will be produced in the robotic cell assembly line. As this parameter increases, our problem becomes harder to be solved by the commercial solvers DICOPT or CPLEX. The second one is the ratio of the profit contribution of the models to the station investment cost, denoted as $\frac{PR_h}{V_j}$. This ratio is very important to evaluate the tradeoff between the revenue and the number of stations that will be used in the assembly line. We set two levels as low and high for both of these factors. When the number of models is at its low level we use $g = 2$ and when it is at its high level, we use $g = 4$. We use a constant cost of \$180000 for setting up one station in the assembly line, $V_j = 180000$ for $j = 1, 2, \ldots, m$ where m is an upper bound for the number of stations. When the ratio $\frac{PR_h}{V_j}$ is at its low level, PR_h is a uniform random number from the interval [5,7]. When this ratio is at its high level, the interval from which we select the values for PR_h is [9,11]. Using the combinations of these two factors at low and high level, we have four main clusters for our runs in which these two factors have low-low, low-high, high-low and high-high levels. We take 5 replications for each cluster so a total of 20 different runs are taken.

The value of the parameter n_h, the number of spot welding operations required for model h, is a uniform random number from the interval [8,15]. The value of the parameter W_{hi}, the number of spot welds required to perform operation i of model h, is also a uniform random number from the interval [6,15]. In all the runs, we used $m = 6$ for the upper bound on the number of stations that can be used in the assembly line. We used a constant value of 3 seconds for the parameter $\tau_j, j = 1, \ldots, 6$ and a constant value of \$10 for $C_j, j = 1, \ldots, 6$ for the cost of welding tools. The values of the parameters $B_j, j = 1, \ldots, 6$ are uniform random numbers from the interval [3200,4000]. We also used constant values of 1 seconds and 2 seconds for the time required for a robot in a station to reach to the position to perform an operation and the time required to perform a single spot weld, respectively ($\alpha = 1, \beta = 2$).

When the number of models is at the low level, we use $\theta_1 = 150000$ and $\theta_2 = 75000$ and at the high level, we use $\theta_1 = 100000$, $\theta_2 = 50000$, $\theta_3 = 50000$ and $\theta_4 = 25000$ for the values of expected demands. We set the value of the expected profit for each excess production of model h as $PR_h^{\epsilon} = \lceil PR_h \cdot 0.7 \rceil$.

In order to set the values of the cycle time lower bound (γ_h^L), cycle time upper bound (γ_h^U) and target cycle time (γ_h) parameters for a particular model h, we use the following procedure. First, we calculate $\bar{t}_{hi}$, the average time to perform one spot welding operation of model h. We round $3 \cdot \bar{t}_{hi}$ to the nearest integer and use this value for γ_h^L. We round $4 \cdot \bar{t}_{hi}$ to the nearest integer and use this value for γ_h^U. The value of the parameter γ_h is a uniform random number from the interval $[3.3 \cdot \gamma_h^L, 3.6 \cdot \gamma_h^U]$.

In all of our runs, we assumed that a welding operation can be assigned to any station ($a_{hij} = 1$ for $h = 1, \ldots, g$, $i = 1, 2, \ldots, n_h$, $j = 1, 2, \ldots, 6$). Generally, there are additional subcomponents to be assembled to the body components inside the robotic cell. These subcomponents should be assembled in a particular order. Therefore, assembling subcomponents causes a set of operations to precede some other sets of operations. Hence, for the precedence relations of welding operations of a model h, we use the $(n_h \text{x} n_h)$ upper left submatrix of the following precedence matrix.

Table 6.1: Precedence matrix for computational study runs

$i\backslash^k$	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	0	0	0	1	1	1	0	0	0	0	0	0	0	0	0
2	0	0	0	1	1	1	0	0	0	0	0	0	0	0	0
3	0	0	0	1	1	1	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0
5	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0
6	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0
8	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0
9	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0
10	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1
11	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1
12	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1
13	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
14	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Having defined the data set that we use in our runs, we now continue with some numerical results of our computational study. First, we solve each run with Algorithm 1 and Algorithm 2. Then, we run DICOPT and CPLEX for a maximum time limit of 2 hours (i.e., 7200 seconds) in order to find solutions for NLMIP and MIP versions of our instances, respectively. We give the solutions for our original problems under the names DICOPT 1 and MIP 1. Finally, in order to improve DICOPT and CPLEX performances, we provide the best Algorithm 1 solutions as initial solutions to our instances and give the results for these problems under the names DICOPT 2 and MIP 2.

We start our analysis with Algorithm 1 and Algorithm 2 solutions. The objective function values found by Algorithm 1 and Algorithm 2 can be seen in Table 6.2. The percentage of increase in the Total Profit values is calculated by the following formula:

$$I_{Alg1-Alg2} = \frac{(\text{Algorithm 2 value}) - (\text{Algorithm 1 value})}{(\text{Algorithm 1 value})} * 100$$

Table 6.2: Total Profit Values for Algorithm 1 and Algorithm 2

Factor						
h	$\frac{PR}{V_j}$	*Replication*	*Run #*	*Algorithm* 1	*Algorithm* 2	$I_{Alg1-Alg2}$
		1	1	460055	507430	10.3
		2	2	460200	557526	21.2
L	L	3	3	699150	699150	0.0
		4	4	354909	399544	12.6
		5	5	583862	622650	6.6
		1	6	1443553	1493919	3.5
		2	7	1232202	1339032	8.7
L	H	3	8	1351146	1509570	11.7
		4	9	1321299	1321299	0.0
		5	10	1882792	1882792	0.0
		1	11	668407	713824	6.8
		2	12	478329	583417	22.0
H	L	3	13	401048	453502	13.1
		4	14	586373	634184	8.2
		5	15	657755	723984	10.1
		1	16	1516720	1558088	2.7
		2	17	1663509	1725291	3.7
H	H	3	18	1200298	1272524	6.0
		4	19	1042092	1130496	8.5
		5	20	1205923	1210876	0.4

In order to obtain the solutions provided in Table 6.2, we use the following procedure: First, we start with the original cycle time upper bounds for the models to be produced in the assembly line and obtain the cycle times, the Total Profit value and the number of stations required in the solution by using Algorithm 1. In the second step, by using the surrogate problem, we check whether the obtained cycle times are achievable with less number of stations. If they are, we calculate the corresponding Total Profit Value. If they are not, we run the surrogate problem to obtain a better allocation of operations and lower cycle times with the same number of stations obtained from the solution of Algorithm 1. The new allocation of operations is the Algorithm 2 solution and we calculate the corresponding Total Profit value for the new solution. By setting $\widehat{\gamma}_h^U = f_h - 1$, we continue this process until we obtain cycle times that are less than or equal to the cycle time lower bounds for each model. Table 6.3 is a summary of this procedure for run #1.

The best Total Profit value found by Algorithm 1 is 460055. In this solution, the revenue gained as the sum of the individual profits of the parts produced is

Table 6.3: Calculation of Total Profit Values for Algorithm 1 and Algorithm 2

Cycle time upper bounds	*Algorithm 1 cycle times*	*Algorithm 1 Total Profit values*	*Number of stations (m_y)*	*Achievable with $m_y - 1$ stations?*	*Algorithm 2 cycle times*	*Algorithm 2 Total Profit values*
$88 - 88$	$87 - 86$	396069	4	*No*	$76 - 83$	**507430**
$86 - 85$	$81 - 83$	453340	4	*No*	$76 - 83$	507430
$80 - 82$	$76 - 74$	378247	5	*No*	$60 - 68$	479839
$75 - 73$	$74 - 70$	411340	5	*No*	$60 - 68$	479839
$73 - 69$	$68 - 68$	**460055**	5	*No*	$60 - 68$	479839
$67 - 67$	$64 - 60$	308939	6	*No*	$60 - 55$	308939

1522055, a total station cost of 900000 for setting up 5 stations and a total tooling cost of 162000 are incurred. However, the best Algorithm 2 solution requires 4 stations to be set up in the assembly line. In this solution, a revenue of 1357030 is gained but the total station cost and the total tooling cost decrease to 720000 and 129600, respectively. Although less revenue is gained, a larger Total Profit value is achieved as a result of the decreases in the cost terms.

As we can see in Table 6.2, except for three runs, by the help of the surrogate problem, we obtain a significant increase by Algorithm 2 in terms of the objective function value with respect to Algorithm 1 solutions. More importantly, as we have mentioned before, it is very difficult to find even a feasible solution for our problem. However, we are able to find good feasible solutions to our instances by Algorithm 1 and improve these solutions by Algorithm 2. The results for DICOPT 1 and MIP 1 instances are given in Table 6.4. The best possible values given in Tables 6.4 and 6.5 are the Total Profit values of the most promising nodes in the Branch-and-Bound trees of the corresponding runs.

The results shown in Table 6.4 give us insights about the difficulty of our problem. DICOPT, a nonlinear solver in GAMS, can not find any feasible solution to any of our instances in a time limit of 2 hours. In Table 6.4 and the rest of the tables given in this chapter, NS stands for 'no solution'. CPLEX also fails to find even a feasible solution to our instances with 4 models to be produced in the assembly line. In 9 of the runs with 2 models, CPLEX can find feasible solutions. However, except run # 8, all of these solutions are worse than Algorithm 1 solutions in terms of objective function values. Furthermore, Algorithm 2 finds

Table 6.4: Total Profit Values for DICOPT 1 and MIP 1

Run #	*DICOPT* 1	*MIP* 1	*Best Possible*	*CPU Time*
1	*NS*	120319	862639	7200
2	*NS*	257418	817886	7200
3	*NS*	*NS*	*NS*	7200
4	*NS*	283783	731878	7200
5	*NS*	396215	949257	7200
6	*NS*	1339492	1825492	7200
7	*NS*	1199099	1605362	7200
8	*NS*	1439515	1796122	7200
9	*NS*	598773	1671504	7200
10	*NS*	1715819	2117107	7200
11	*NS*	*NS*	*NS*	7200
12	*NS*	*NS*	*NS*	7200
13	*NS*	*NS*	*NS*	7200
14	*NS*	*NS*	*NS*	7200
15	*NS*	*NS*	*NS*	7200
16	*NS*	*NS*	*NS*	7200
17	*NS*	*NS*	*NS*	7200
18	*NS*	*NS*	*NS*	7200
19	*NS*	*NS*	*NS*	7200
20	*NS*	*NS*	*NS*	7200

solutions with larger objective function values for all the instances for which CPLEX can find a feasible solution in a time limit of 2 hours (or 7200 seconds).

We continue our analysis with DICOPT 2 and MIP 2 solutions, in which we provide the Algorithm 1 solutions as initial solutions to our instances. By providing initial feasible solutions to DICOPT and CPLEX for our runs, we aim to reduce the search space and search time of our runs. Initial solution values will constitute a lower bound for our objective function value and we expect to obtain better results than those that are achievable by DICOPT 1 and MIP 1. Table 6.5 shows DICOPT 2 and MIP 2 solutions to our instances.

As we can see from Table 6.5, except run # 8, MIP 2 solutions are better than MIP 1 solutions, which were given in Table 6.4. In other words, providing the Algorithm 1 solutions of our runs to CPLEX as initial solutions helps us to improve CPLEX results in 95% of our problem instances. As a result of the complexity and difficulty of our problem, still none of the solution values found by MIP 2 is better than the solution that we are able to find by Algorithm 2.

Table 6.5: Total Profit Values for DICOPT 2 and MIP 2

Run #	*DICOPT 2*	*MIP 2*	*Best Possible*	*CPU Time*
1	*NS*	478044	792832	7200
2	*NS*	536910	760823	7200
3	*NS*	699150	943616	7200
4	*NS*	354909	687143	7200
5	*NS*	583862	951209	7200
6	*NS*	1465125	1754679	7200
7	*NS*	1312737	1568786	7200
8	*NS*	1401586	1718384	7200
9	*NS*	1321299	1615406	7200
10	*NS*	1882792	2117928	7200
11	*NS*	668407	957151	7200
12	*NS*	478329	809448	7200
13	*NS*	401048	846789	7200
14	*NS*	586373	894127	7200
15	*NS*	657755	954369	7200
16	*NS*	1516720	1783367	7200
17	*NS*	1663509	1999991	7200
18	*NS*	1200298	1598373	7200
19	*NS*	1042092	1460130	7200
20	*NS*	1205923	1549724	7200

Another important improvement that we achieve by providing initial solutions is that we can significantly reduce the gap between the best possible values and the solutions that CPLEX find in 2 hours. One of the main reasons for these improvements is that except one instance, Algorithm 1 solutions are better than MIP 1 solutions in terms of objective function values. Another reason is that providing an initial solution help to reduce the search space for our instances. Table 6.6 shows MIP 1 and MIP 2 gaps, which are the differences between the best possible values and the Total Profit values of the best integer solutions found for the corresponding runs. The percentage of decrease in gaps is calculated by the following formula:

$$G_{MIP1-MIP2} = \frac{(\text{MIP 1 gap}) - (\text{MIP 2 gap})}{(\text{MIP 1 gap})} * 100$$

Providing initial solutions to CPLEX helps to obtain an average of 42.2% decrease in the gaps. Such decreases in the gaps between the best possible values and the solutions found help us to ensure that we get closer to the optimal values by providing initial solutions to CPLEX.

Table 6.6: Gap values for MIP 1 and MIP 2

Run #	MIP 1 *gap*	MIP 2 *gap*	$G_{MIP1-MIP2}$
1	742320	314788	57.6
2	560468	223913	60.1
3	–	244466	–
4	448095	332234	25.9
5	553042	367347	33.6
6	486000	289554	40.4
7	406263	256049	37.0
8	356607	316798	11.2
9	1072731	294107	72.6
10	401288	235136	41.4
11	–	288744	–
12	–	331119	–
13	–	445741	–
14	–	307754	–
15	–	296614	–
16	–	266647	–
17	–	336482	–
18	–	398075	–
19	–	418038	–
20	–	343801	–

Up until now, we were able to show that in all of the instances, we can find better solutions with Algorithm 2 than the ones that can be found by CPLEX with or without initial solutions. While finding better solutions is still meaningful, there is also another point that we should take into consideration. This another important point is the time that we spent to solve our instances. In Table 6.7, we give a comparison of the objective function values and solution times that we spent to solve our instances between Algorithm 2 and CPLEX. Although very few, there may be instances in which MIP 1 can find a solution that is better than the one that can be found by MIP 2 in terms of Total Profit value. Therefore, while comparing Algorithm 2 and CPLEX, we take the best of MIP 1 and MIP 2 solutions for the corresponding runs. The percentage of the difference in Total Profit Values between CPLEX and Algorithm 2 solutions is calculated by the following formula:

$$D_{CPLEX-Alg2} = \frac{(\text{Algorithm 2 value}) - (\text{Best of MIP 1 and MIP 2 value})}{(\text{Best of MIP 1 and MIP 2 value})} * 100$$

As we mentioned before, we run DICOPT and CPLEX for 2 hours for each

Table 6.7: Total Profit Values and CPU Times for CPLEX and Algorithm 2

	Best of	*MIP*		*Alg* 2	
Run #	*MIP* 1 *and MIP* 2	*CPU Time*	*Alg* 2	*CPU Time*	$D_{CPLEX-Alg2}$
1	478044	7200	507430	3.5	6.2
2	536910	7200	557526	2.7	3.8
3	699150	7200	699150	5.6	0.0
4	354909	7200	399544	3.0	12.6
5	583862	7200	622650	3.8	6.6
6	1465125	7200	1493919	2.8	2.0
7	1312737	7200	1339032	2.7	2.0
8	1439515	7200	1509570	2.3	4.9
9	1321299	7200	1321299	4.0	0.0
10	1882792	7200	1882792	2.2	0.0
11	668407	7200	713824	8.4	6.8
12	478329	7200	583417	13.4	22.0
13	401048	7200	453502	263.6	13.1
14	586373	7200	634184	7.1	8.2
15	657755	7200	723984	14.6	10.1
16	1516720	7200	1558088	4.2	2.7
17	1663509	7200	1725291	8.4	3.7
18	1200298	7200	1272524	32.9	6.0
19	1042092	7200	1130496	363.3	8.5
20	1205923	7200	1210876	214.6	0.4

of the DICOPT 1, MIP 1, DICOPT 2 and MIP 2 versions of our 20 runs. By Algorithm 2, we are able to find solutions that are on the average 6.0% better than the corresponding best of MIP 1 and MIP 2 solutions in terms of the Total Profit value. In addition to finding better solutions to our instances, Table 6.7 shows that Algorithm 2 performs much better in terms of CPU times with respect to DICOPT or CPLEX. A relatively large number of stations required to obtain a solution y found by Algorithm 1 increases the search space and search time of our surrogate problem, resulting in relatively greater CPU times in some of the runs (i.e., runs 13, 19 and 20).

In order to find the optimal Total Profit values, we abolished the CPU time restriction and ran CPLEX on a computer with 8 GB memory, Xeon 2.66 GHZ CPU and Linux operating system for the first MIP 2 runs of the first two clusters, in which the number of models is equal to 2 and the problems are relatively easier to be solved by CPLEX. However, after 72660 seconds (20 hours and 11 minutes) and 45840 seconds (12 hours and 44 minutes) for the runs 1 and 6 respectively, CPLEX terminated due to 'out of memory' error since the tree sizes grew to 4.5 GB. The best integer solution found for run 1 was 507430, which is also the Total Profit value that we found by Algorithm 2. The upper bound for the Total Profit value for run 1 was 536671, which implies a 5.7% of gap between the best solution found and the best possible solution in terms of the Total Profit value. The best solution found for run 6 had a Total Profit value of 1509221, which is only 1% better than the Total Profit value that we found by Algorithm 2. The upper bound for the Total Profit value for run 6 was 1546606, which implies a 2.5% of gap between the best solution found and the best possible solution in terms of the Total Profit value. Looking at these results and Example 3 provided in section 4.2, we can say that Algorithm 2 is capable of finding very good solutions to instances of our problem. Also, it would probably take several days to solve the MIP versions of our runs to optimality by CPLEX.

As we mentioned before, the values of the two factors, the number of different models to be produced in the assembly line and the the ratio $\frac{PR_h}{V_j}$ play an important role for the Total Profit values of our solutions and the Heuristic, DICOPT and CPLEX performances. Table 6.8 is a summary of the cluster based evaluation of Algorithm 1, Algorithm 2 and CPLEX.

Table 6.8: Solution Analysis for Problem Clusters

$Factor$		$I_{Alg1-Alg2}$			$D_{Alg1-CPLEX}$			$D_{CPLEX-Alg2}$		
h	$\frac{PR}{V_j}$	Min	Ave	Max	Min	Ave	Max	Min	Ave	Max
L	L	0.0	10.1	21.2	0.0	4.1	16.7	0.0	5.8	12.6
L	H	0.0	4.9	11.7	0.0	2.9	6.5	0.0	1.8	4.9
H	L	6.8	12.0	22.0	0.0	0.0	0.0	6.8	12.0	22.0
H	H	0.4	4.3	8.5	0.0	0.0	0.0	0.4	4.3	8.5

When the number of models is at its low level, Algorithm 1 sometimes works as well as Algorithm 2. However, in all of the runs with the number of models at the high level, we obtain better results with Algorithm 2 because as the number of models increase, the total number of operations increase and hence the probability of improving Algorithm 1 allocations by the surrogate problem and obtaining better solutions increase. The improvements are more significant especially in the runs with low $\frac{PR_h}{V_j}$ ratios because the Total Profit values are relatively smaller when $\frac{PR_h}{V_j}$ ratios are at the low level and as a result, improvements on Algorithm 1 solutions in terms of Total Profit values correspond to higher percentages. In addition to this, CPLEX can improve the initial solutions found by Algorithm 1 in 5 of the 10 runs with low number of models and fails to improve any of the solutions found by Algorithm 1 for the runs with high number of models. Again, CPLEX improvements on Algorithm 1 solutions are more significant when the ratio $\frac{PR_h}{V_j}$ is at its low level. Moreover, except for 3 runs, Algorithm 2 finds better solutions than CPLEX in terms of the Total profit values and the differences between Algorithm 2 and CPLEX solutions are more significant in the runs with low $\frac{PR_h}{V_j}$ ratios. The reason of why Algorithm 2 can not find a better solution than CPLEX solutions in 3 of the runs is that in these 3 runs, Algorithm 1 and Algorithm 2 Total Profit values are the same and that CPLEX takes Algorithm 1 solutions as initial feasible solutions.

In summary, in this chapter of our study, we tested the performance of our heuristic algorithm on some instances of our problem that are created randomly. The results of the comparison between our heuristic and GAMS solvers DICOPT and CPLEX showed that our heuristic is very efficient in terms of the CPU time and the quality of the solutions found. The next chapter of this thesis is devoted to concluding remarks.

Chapter 7

Conclusion

In this study, we considered a mixed-model assembly line design problem with the objective of maximizing the total profit that would be gained from the assembly line. In the assembly line, different models are produced in an intermixed sequence. Due to the similarities between the production processes of different models, setup times are not present. Instead of assuming continuous production, we dealt with designing an assembly line that works 24 hours a day in three 8-hour shifts. We considered a 10-minute break between the shifts and a 30-minute lunch break and two 10-minute tea breaks in each shift. Considering limited tool lives for the tools used for production in the assembly line gives rise to the necessity of determining the replacement times of the tools. In many industries, tool change related assembly line stoppages result in loss of production and so are very costly. In our study, we eliminate such a problem by restricting tool changes to be performed only in the aforementioned breaks.

We first formulated our assembly line design problem as a Nonlinear Mixed Integer Programming (NLMIP) model and tried to solve our problem with DICOPT. Due to the size and difficulty of our problem, DICOPT failed to find a feasible solution to any one of the generated instances, which we think closely represent real life problems that we have observed during our collaboration with the automotive industry. Therefore, we linearized our problem and tried to solve the Mixed Integer Programming (MIP) version of our problem by CPLEX. However, CPLEX also failed to solve our problem in the preset time limit and could not find even a feasible solution for several instances.

Upon not obtaining optimal solutions from DICOPT and CPLEX (we may not even have a feasible solution for larger problem instances), we developed a heuristic algorithm to obtain a set of feasible solutions. Finding alternative solutions rather than a single solution allows us to perform a search for the best solution. In addition to this, we improved our heuristic by incorporating a surrogate problem and obtained a stronger heuristic, which performed better in terms of objective function values of the solutions found. We compared the results of the heuristic algorithms given in Sections 4.1 (Algorithm 1) and 4.2 (Algorithm 2) with commercial solvers DICOPT and CPLEX. We also provided the best Algorithm 1 solutions as initial solutions to DICOPT and CPLEX in order to increase their performances. By doing so, we observed that CPLEX could find better solutions in terms of the objective function value and the gap between the objective function values of the best integer solution found and the best possible solution decreased. However, Algorithm 2 outperformed DICOPT and CPLEX in many instances and proved to be very efficient in terms of the solution times and the objective function values of the solutions found.

The existing studies in the literature overlook the limited lives of the tools that are used for production and hence ignore tool change decisions. However, in most of the industries, the tools that are used for production have limited tool lives. As we saw in Chapter 5, assembly lines may suffer from tool change related stoppages unless the tool lives are taken into consideration while allocating operations to the stations in the assembly line. Moreover, real-life assembly line problem formulations may take a very long time to be solved by commercial solvers. Hence, by representing a more realistic production environment and with the efficient solution techniques provided, we hope that our study will be a remarkable contribution to the existing literature.

Our study is the first one to consider tool changes in the assembly lines and to eliminate the tool change related line stoppages. For future research, this study can be extended to the cases in which there may be multiple robots in the stations of the assembly line. Our problem is more applicable on the existing assembly lines, which are subject to model changes. But, if an assembly line is to be set up for the first time, tool selection decision can be added to our problem by incorporating tool selection decision variables. Furthermore, as Gultekin *et al.* [11] did, flexibility can be added to the scope of our study by considering controllable processing times instead of using deterministic processing times for the assembly line operations. In addition to these, by adding assembly line setup times for the models to be produced, our problem can be converted to a multi-model assembly line problem, in which an additional sequencing problem will arise.

Bibliography

[1] R. G. Askin and M. Zhou. A parallel station heuristic for the mixed-model production line balancing problem. *International Journal of Production Research*, 35:3095–3105, 1997.

[2] I. Baybars. An efficient heuristic method for the simple assembly line balancing problem. *International Journal of Production Research*, 24:149–166, 1986.

[3] I. Baybars. A survey of exact algorithms for the simple assembly line balancing problem. *Management Science*, 32:909–932, 1986.

[4] C. Becker and A. Scholl. A survey on problems and methods in generalized assembly line balancing. *European Journal of Operational Research*, 168:694–715, 2006.

[5] J. Bukchin, E. M. Dar-el, and J. Rubinovitz. Mixed-model assembly line design in a make-to-order environment. *Computers and Industrial Engineering*, 41:405–421, 2002.

[6] Y. Bukchin and I. Rabinowitch. A branch-and-bound based solution approach for the mixed-model assembly line-balancing problem for minimizing stations and task duplication costs. *European Journal of Operational Research*, 174:492–508, 2005.

[7] E. Erel and H. Gökcen. Shortest route formulation of mixed-model assembly line balancing problem. *European Journal of Operational Research*, 116:194–204, 1999.

[8] E. Erel, I. Sabuncuoglu, and B. A. Aksu. Balancing of U-type assembly line systems using simulated annealing. *International Journal of Production Research*, 39:3003–3015, 2001.

[9] E. Erel, I. Sabuncuoglu, and H. Sekerci. Stochastic assembly line balancing using beam search. *International Journal of Production Research*, 43:1411–1426, 2005.

[10] K. Fleszar and K. S. Hindi. An enumerative heuristic and reduction methods for the assembly line balancing problem. *European Journal of Operational Research*, 145:606–620, 2003.

[11] H. Gultekin, M. S. Akturk, and O. Karasan. Bicriteria robotic cell scheduling. *Journal of Scheduling*, 11:457–473, 2008.

[12] A. N. Haq, J. Jayaprakash, and K. Rengarajan. A hybrid genetic algorithm approach to mixed-model assembly line balancing. *International Journal of Advanced Manufacturing Technology*, 28:337–341, 2006.

[13] N. V. Hop. A heuristic solution for fuzzy mixed-model line balancing problem. *European Journal of Operational Research*, 168:789–810, 2006.

[14] G. Levitin, J. Rubinovitz, and B. Shnits. A genetic algorithm for robotic assembly line balancing. *European Journal of Operations Research*, 168:811–825, 2006.

[15] S. Matanachai and C. A. Yano. Balancing mixed-model assembly lines to reduce work overload. *IIE Transactions*, 33:29–42, 2001.

[16] P. R. McMullen and P. Tarasewich. Using ant techniques to solve the assembly line balancing problem. *IIE Transactions*, 35:605–617, 2003.

[17] C. Merengo, F. Nava, and A. Pozetti. Balancing and sequencing manual mixed-model assembly lines. *International Journal of Production Research*, 37:2835–2860, 1999.

[18] J. Miltenburg. Balancing and scheduling mixed-model U-shaped production lines. *International Journal of Flexible Manufacturing Systems*, 14:119–151, 2002.

[19] B. Rekiek, P. de Lit, F. Pellichero, T. L'Eglise, P. Fouda, E. Falkenauer, and A. Delchambre. A multiple objective grouping genetic algorithm for assembly line balancing. *Journal of Intelligent Manufacturing*, 12:467–485, 2001.

[20] T. Sawik. Monolithic vs. hierarchical balancing and scheduling of a flexible assembly line. *European Journal of Operational Research*, 143:115–124, 2002.

[21] A. Scholl and C. Becker. State-of-the-art exact and heuristic solution procedures for simple assembly line balancing. *European Journal of Operations Research*, 168:666–693, 2006.

[22] A. Scholl and R. Klein. Balancing assembly lines effectively–a computational comparison. *European Journal of Operational Research*, 114:50–58, 1999.

[23] D. Sparling and J. Miltenburg. The mixed-model U-line balancing problem. *International Journal of Production Research*, 36:485–501, 1998.

[24] P. M. Vilarinho and A. S. Simaria. An ant colony optimization algorithm for balancing mixed-model assembly lines with parallel workstations. *International Journal of Production Research*, 44:291–303, 2006.

[25] W. E. Wilhelm and R. Gadidov. A branch-and-cut approach for a generic multiple-product, assembly-system design problem. *INFORMS Journal on Computing*, 16:39–55, 2004.

Appendix A

Nomenclature

Problem Parameters

V_j: cost of setting up station j, $j \in M$.

n_h: number of operations to be performed to produce model h, $h \in G$.

W_{hi}: number of spot welds required to perform operation i of model h, $i \in N_h$, $h \in G$.

B_j: total number of spot welds such that the welding tool in station $j \in M$ can process.

C_j: cost of the welding tool in station$j$, $j \in M$.

θ_h: yearly expected demand for model h, $h \in G$.

γ_h: target cycle time for model h to meet the yearly expected demand θ_h, $h \in G$.

γ_h^L: lower bound for the actual cycle time of model h, $h \in G$.

γ_h^U: upper bound for the actual cycle time of model h, $h \in G$.

PR_h: profit contributed by each component of model h produced, $h \in G$.

PR_h^ϵ: profit contributed by each excess component of model h produced, $h \in G$.

$a_{hij} = 1$, if operation $i \in N_h$ of model $h \in G$ can be assigned to station $j \in M$; and 0, otherwise.

$p_{hik} = 1$, if operation $i \in N_h$ of model $h \in G$ precedes operation $k \in N_h$ of model $h \in G$; and 0, otherwise.

t_{hi}: the time required to perform operation $i \in N_h$ of model $h \in G$.

D_q: available tool change time in tool change period q, $q = 1, 2, \ldots, U$.

K: time required to prepare for welding tool changes.

μ: time required to replace a single welding tool.

α: time required for a robot to reach to the position to perform an operation.

β: time required to process a single spot weld.

τ_j: time required for the robot in station $j \in M$ to begin and finalize the allocated operations.

ψ_h: proportion of time that should be allocated to the production of model h.

ρ: constant used to convert total tooling cost to yearly tooling cost.

η: constant used to convert total station cost to yearly station cost.

Decision Variables

R_{jq}: remaining number of spot welds such that the welding tool in station j can process after tool change period q, $j = 1, 2, \ldots, m$, $q = 1, 2, \ldots, U$.

f_h: actual cycle time for model h.

σ_j: 1, if station $j \in M$ is used in the assembly line; and 0, otherwise.

$x_{hij} = 1$, if operation $i \in N_h$ of model $h \in G$ is assigned to station $j \in M$; and 0, otherwise.

$z_{jq} = 1$, if tool in station $j \in M$ is changed in tool change period q, $q = 1, 2, \ldots, U$; and 0, otherwise.

ω_h: variable that is used to linearize $\frac{1}{f_h}$.

$AO_h, BO_h, \lambda_h^1, \lambda_h^2, y_h$: variables used to linearize the objective function.

e_{hij}: variable used to linearize $x_{hij} \cdot \omega_h$.

F: a large number used for linearization.

O_{jq}: variable used to linearize $R_{j(q-1)} \cdot z_{jq}$.

π_{hijq}: variable used to linearize $e_{hij} \cdot z_{jq}$.

VITA

Adnan Tula was born on March 3, 1985 in Isparta, Türkiye. He received his high school education at Antalya Özel Mahmut Celal Ünal Fen Lisesi, Türkiye. He has received his B.S. degree from the Department of Industrial Engineering, Bilkent University in 2007. Since September 2007, he has been working with Prof. Selim Aktürk for his graduate study at the same department.