A DECISION SUPPORT SYSTEM FOR PRODUCTION PLANNING AND SCHEDULING IN A SMART FACTORY ENVIRONMENT

HAKTAN CİNKILIÇ

BOĞAZİÇİ UNIVERSITY

2017

A DECISION SUPPORT SYSTEM FOR PRODUCTION PLANNING AND SCHEDULING IN A SMART FACTORY ENVIRONMENT

Thesis submitted to the

Institute for Graduate Studies in Social Sciences

in partial fulfillment of the requirements for the degree of

Master of Arts

in

Management Information Systems

by

Haktan Cinkılıç

Boğaziçi University

2017

DECLARATION OF ORIGINALITY

I, Haktan Cinkılıç, certify that

- I am the sole author of this thesis and I have fully acknowledged and documented in my thesis all sources of ideas and words, including digital resources, which have been produced or published by another person or institution;
- this thesis contains no material that has been submitted or accepted for a degree or diploma in any other educational institution;
- this is a true copy of thesis approved by my advisor and thesis committee at Boğaziçi University.

Signature……………………………………..
Date……02.05.2017……………

ABSTRACT

A Decision Support System for Production Planning and Scheduling in a Smart Factory Environment

With the industry 4.0 idea that emerged recently, manufacturing systems have been enhanced by advanced manufacturing technologies. These smart manufacturing systems can respond faster to the changes in the production plans which are mostly due to the updates in customer order quantities, order due dates, unexpected machine breakdowns, material supply problems, etc. This study aims to support the dynamic structure of smart factories by providing an efficient, effective and flexible platform for production planning and scheduling. In most of the studies in the literature, production control and scheduling plans have been handled separately and iteratively due their computational complexities. As an improvement, in this thesis a mixed integer quadratic programming (MIQP) model is developed to optimize the integrated production plans and daily schedules with minimum total cost. Then, this optimization model is linearized and a mixed integer linear programming (MILP) model is generated to improve its computational performance. Finally, the optimization model is embedded into a web based decision support system (DSS) together with a database and user interface to provide an efficient, effective and flexible decision making environment in a smarter cyber-physical system. The DSS is verified with different test scenarios and its computational performance is measured.

ÖZET

Akıllı Fabrika Ortamında Üretim Planlama ve Zaman Çizelgeleme İçin Bir Karar Destek Sistemi

Son dönemlerde ortaya çıkan Endüstri 4.0 vizyonuyla beraber, üretim sistemleri ileri üretim teknolojileriyle zenginleştirildi. Bu akıllı üretim sistemleri değişken müşteri taleplerinden, esnek teslim tarihlerinden, teknik arızalardan, ürün tedarik problemleri vb. sebeplerden kaynaklanan üretim planı değişikliklerine daha hızlı cevap verebiliyorlar. Bu çalışma akıllı fabrikaların dinamik yapısını desteklemek amacıyla üretim planlama ve zaman çizelgeleme için verimli, etkili ve esnek bir platform sunmayı hedefliyor. Literatürdeki çalışmaların birçoğunda, üretim planlama ve zaman çizelgeleme problemleri hesaplama zorluğundan dolayı ayrı ayrı ele alınıyor. Bir geliştirme olarak, bu tez çalışmasında üretim planlama ve zaman çizelgeleme problemini birlikte ele alan ve maliyetini en aza indirecek bir Karışık Tamsayı Karesel İzlenceleme modeli geliştirildi. Ardından, bu model, hesaplama performansını artırmak amacıyla doğrusallaştırıldı ve bir Karışık Tam Sayılı Doğrusal İzlenceleme modeli oluşturuldu. Son olarak, bu optimizasyon modeli akıllı bir siber-fiziksel sistemde verimli, etkili ve esnek bir karar verme ortamı yaratmak amacıyla bir web tabanlı, veri tabanıyla desteklenmiş bir Karar Destek Sistemi (KDS) içerisine yerleştirildi. Bu KDS birçok test senaryosuyla test edildi ve performansı ölçüldü.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABBREVIATIONS

APP	Aggregate Production Planning
CP	Constraint Programming
CSV	Comma-separated Values
DP	Detailed Planning
DSS	Decision Support System
ERP	Enterprise Resource Planning
FMCG	Fast-moving Consumer Goods
GAMS	General Algebraic Modeling System
GUI	Graphical User Interface
HDSS	Hierarchical Decision Support System
HPP	Hierarchical Production Planning
IoT	Internet of Things
IP	Integer Programming
JIT	Just In Time
LP	Linear Programming
MA	Memetic Approach
MIP	Mixed Integer Programming
MIQP	Mixed Integer Quadratic Programming
MRP	Material Requirement Planning
NP-Hard	Non-deterministic Polynomial-time Hard
PMFRS	Parallel Machine Flexible Resources Scheduling
RH	Rolling Horizon
RHP	Rolling Horizon Procedure

RP	Relaxed Planning
RDMS	Relational Database Management System
SQL	Structured Query Language
SP	Stored Procedures

CHAPTER 1

INTRODUCTION

Smart manufacturing concept has emerged in the last decade with the targets provided by the 4th Industrial revolution idea. Today, companies are trying to combine information technologies, engineering and production to manage business operations more efficiently. In the smart factory structure, cyber-physical systems are main controllers of the system that create a copy of physical world and make decisions. Over the Internet of Things (IoT), the data about manufacturing processes is gathered using RFID tags, censors, actuators and mobile phones. Then, actuators are used to adjust the manufacturing processes as controlled by the human controllers or digital agents that continuously observe the performance of the system via IoT. As Davis et al. (2012) stated smart manufacturing aims to take advantage of advanced information and manufacturing technologies to enable flexibility in physical processes to address a dynamic and global market. In this regard, this study aims to support dynamic structure of smart factories with the help of quick responses to the planned or unplanned events in manufacturing phases.

In a classic manufacturing system, sales and marketing departments provide demanded quantities to manufacturing department. Then, a planner organizes the resources including labor, raw material, machines etc. and a production planner determines the production schedule according to due dates of demands by aiming minimum cost. However, in today’s dynamic world, demanded quantities and due dates change frequently. Changes in the competitive environment due to promotions at the substitute products, or a recently launched new product, or the emergency orders from the retailers that avoid keeping inventories might be the reasons of such fluctuations. Moreover, internal uncertainties might arise mostly due to unexpected events like machine breakdowns or material deficiencies. In this regard, manufacturers have to find a way to update plans quickly and effectively. Especially in sectors like fast-moving consumer goods (FMCG) satisfying the customer need is crucial because the products are substitutable due to their low product differentiation between brands.

For creating resilient and flexible manufacturing systems, production planning and scheduling play critical roles because as Yang et al. (2001) state, those tasks strongly affect profitability of manufacturing a product, resource utilization and product delivery time. In this thesis study we generate an integrated mathematical model to optimize the inventory control and production scheduling problem in a detergent manufacturing system. The original formulation turns out to be a mixed integer quadratic programming (MIQP) optimization problem which can be used to solve only moderate size problems. Later this model is improved by linearizing all nonlinear constraints and the final mixed integer linear programming (MILP) formulation can solve moderate and large scale problems. The generated MILP is embedded into a DSS by integrating the model base with a database and graphical user interface.

The originality of the thesis comes from the fact an “integrated” formulation is provided for the inventory control and production scheduling problem which is very rare in the literature due to its computational complexity. Secondly the generated formulation is linearized by using mathematical techniques without introducing any further assumptions that endanger the validity of the generated model. Thirdly the generated model is embedded into a DSS to generate an effective, efficient and flexible decision making environment for an Industry 4.0 manufacturing system.

The organization of this thesis study is as follows: In the next section we provide a comprehensive review of the studies in the literature on scheduling problems and integrated models for production and scheduling. The definition of the problem and generation of the integrated production and scheduling model is provided in Section 3. In Section 4, a DSS is developed which includes the optimization model as the model base, a database and a user interface. In Section 5, the performance of the DSS and the findings are summarized, and future research directions are given.

CHAPTER 2

LITERATURE REVIEW

2.1 Scheduling

In this section, the studies related with scheduling and integrated production planning are reviewed. In recent years, researchers have been interested in developing models to handle production planning and scheduling together because as Tan (2000) stated typical scheduling objectives such as minimizing makespan, maximizing equipment utilization, etc., could be significantly improved as the result of integration of these two important manufacturing system functions. Yet there are many studies which separately handle production planning and scheduling decisions. In this regard, we will firstly focus on different types of scheduling problems with different solution types. Then, we will review the studies which combines production planning and scheduling.

Fierce competition atmosphere forces companies to deliver customized products in shorter lead times. Hence, production planning and scheduling become more crucial. As the current schedule practice does not meet the demands, more researchers are interested in scheduling. The first studies about scheduling appeared in 1950s with the paper of Johnson (1954). Since then, thousands of papers dealing with the most diverse aspects of scheduling problems have appeared and these can be classified in several ways. We would like to approach the scheduling problems with respect to the nature of setups, job shop environments and solution methodologies for the developed models. Before starting articles, those classifications and our subject problem settings will be clarified.

Firstly, according to Ostwald (1992) setup can be defined as a work to prepare the machine, process, or bench for product parts or the cycle. The setup operations had been considered as negligible or a part of processing time for a long time. However, the importance of setup has been investigated in several recent studies. Accordingly, we can mention two types of setups, namely, sequence dependent setup and sequence independent setup. In the sequence independent type, setup time depends only on the next job to be processed while in the sequence dependent setup, setup time changes according to both processing job and immediately preceding job. Wilbrecht and Prescott (1969) state that sequence-dependent setup times are significant when a job shop is operated at or near full capacity. Krajewski et al. (1996) analyze the factors that affect the production performance and indicate that regardless of the production system in use, simultaneous reduction of setup times and lot sizes are the most effective ways to reduce inventory levels and improve flexibility in satisfying dynamic customer demands.

Secondly, shop environment is another significant criteria for scheduling problems. Some scheduling problems include just one machine while others include many machines. In addition, machines may be substitutionary or complimentary. If a job can be processed on any of the machines then the system is called parallel line production system. And lastly, if all parallel machines are the same in terms of processing time and cost then this system called identical parallel lines. Otherwise, it is called non-identical parallel lines.

Lastly, many different solution techniques are used to optimize scheduling problems which are difficult to solve in most cases. Parker et al., (1977) stated that even parallel line scheduling problem -which is indeed a kind of vehicle routing problem- comes out to be NP hard. Various heuristics types, linear programming, constraint programming are widely used in order to solve those problems in polynomial time.

In scheduling problems, the three-field notation developed by Lawler et al. (1993) is widely used. In this type of notation, the first field is used to define the shop type, the second field is used for setup information and third field stands for base performance criteria. Following this line of approach, in our review, we explore the studies with sequence dependent setup times, shops with identical parallel machines or else with non-identical parallel machines, and performance criteria changes according to the subject setting. In chronological order, we introduce the following studies:

As Allahverdi et al. (1999) stated in their comprehensive review, Marsh and Montgomery (1973) did one of the first studies on sequence dependent parallel line scheduling problems. They developed a heuristic method and used it for both identical and non-identical machines. Deane and White (1975) extended Marsh and Montgonery's (1993) study by adding workload balancing. They used a branch and bound method algorithm to minimize sequence dependent setup cost. Geoffrion and Graves (1976) developed a quadratic assignment formulation by taking inspiration from the magazine-bindery scheduling problem of Balas et al. (1973). Dearing and Henderson (1982, 1984) developed a linear programming model and offered a solution. Then, Sumichrast and Baker (1987) extended their job by developing a heuristic called MINSET. They aimed to minimize the number of changeovers and compared the results of Dearings' (1984) LP technique and MINSET.

Guinet (1990) studied on a textile production system which contains non-identical parallel processors. He used a four-step solution with graph theory algorithms in order to solve the problem. Guinet and Dussauchoy (1993) also studied on an identical parallel line scheduling problem by rewriting the model developed earlier (Guinet 1984). The basis of rewritten model was to represent routing problem as an assignment problem in which machine redeployment is allowed. They extended the Hungarian method principles which were used by Guinet (1984). After assignment, their heuristic method minimized the setup costs by job permutations.

Ovacık and Uzsoy (1995) developed a class of heuristics methods in order to minimize maximum lateness on identical parallel machines with sequence dependent setup times and dynamic job arrivals. The uniqueness of this Ovacık and Uzsoy's (1995) study stemmed from dynamic jobs that arrived over time. This dynamism motivated them to use Rolling Horizon Procedures (RHP). Ovacık and Uzsoy (1995) stated that in RHP, at each point, a sub problem was solved by using forecasts that were predicted to happen for a certain period in the future. The RHPs used in this study were the extensions of single machine RHPs which were developed by Ovacik and Uzsoy (1994). The results showed that their method performed 69% better than Earliest Due Date dispatching rule and 38% better than Earliest Due Date coupled with local search.

Simulated Annealing is one of the most commonly used heuristic technique in scheduling problems. Lee and Pinedo (1997) applied a three phase heuristic by using simulated annealing in order minimize the sum of the weighted tardiness. In the first phase, the method estimated the makespan. In the second phase, a sequence was constructed by using an extended version of Lee et al.'s (1997) Apparent Tardiness Cost with Setups technique. At the final step, simulated annealing was applied. Various neighborhood structures and acceptance probabilities were tested. Until 60 jobs, the method reached the optimal solution. However, after 60 jobs no significant improvement could be obtained.

Heady and Zhu (1998) studied on a Just-In-Time (JIT) production environment problem which aimed to reduce inventories, produce defect free products and improve response times. In JIT philosophy, both earliness and tardiness incur costs. Therefore, the optimal schedule should be organized to close all jobs at their due dates. Hence, in the first stage of their algorithm a heuristic was developed for a single machine problem and applied to each machine. Then, this heuristic was built into a machine assignment heuristic. The results proved that newly developed heuristic outperformed the other heuristics and linear algorithms. Researchers noted that computational time and accuracy were the points to be improved.

In another study which is done by Vignier et al. (1999), there were two non-identical machines. Jobs could be performed in both types of machines. However, required time and costs were different. They first found a feasible schedule and then developed a heuristic to find the schedule with minimum cost. In order to solve this problem a three phase hybrid heuristics was proposed. The first phase was based on an iterative heuristic, the second on a genetic algorithm and the last one on a branch and bound for post optimization. (Vignier et al., 1999). The results outperformed the previous studies because the genetic algorithm allowed to visit a larger set of feasible solutions than with the Earliest Due Date method.

Radhakrishnan and Ventura (2000) applied another simulated annealing solution for the problem of minimization of earliness and tardiness of identical parallel machine sequence dependent setup times with non-common due dates. In order to enhance the quality of solutions, Radhakrishnan and Ventura developed a heuristic to identify the starting solution of simulated annealing which improved the performance of small-sized problems by 17%, medium-sized problems by 42%, and large-sized problems by 33%. The results were compared with linear programming model which was solved by using CPLEX v 4.0.7. Small sized problems were optimally solved by CPLEX in a long CPU time compared to simulated annealing method. Later, another heuristic called Squeaky Wheel Optimization was applied to the same problem by Fend and Lau (2005). They proved that their method outperformed Radhakrishnan and Ventura (2000). Hiraishi et al. (2002) also studied on the same settings but they aimed to maximize the number of jobs that were completed on their due dates. They showed that this problem could be solved in polynomial time in certain conditions and NP hard in some other cases.

Mendes et al. (2002) considered the problem minimizing makespan for identical parallel lines scheduling. They compared the success of tabu search and a new memetic approach (MA) where the latter was a combination of genetic algorithms with local search procedures. Comparisons were done in two ways. For small sized instances, the methods were compared according to the optimal solutions (or lower bounds) obtained by a dichotomous search scheme. For larger instances, the comparisons were done based on benchmarks that were the best-known solutions obtained by a long time execution of the tabu search implementation. For the tabu search, they used the 3 phase method that was developed by Franca et al. (1996). Initial solution was obtained by assigning all jobs to the random processors. Second phase was a local search heuristic to reach a better solution by moving jobs between different processors. And third phase was a classical tabu search based on changing sequence in order to achieve an improved solution. In the memetic algorithm approach, all individuals of the population evolved solutions until they became a local minima of a certain neighborhood, i.e., after the recombination and mutation steps, a local search was applied to the resulting solutions. The results showed that MA performed better when the number of machines is small and the setup times were small compared to processing times. However, with large setup times and many machines, Fast Tabu Search performed better. Another comparative study for the same problem was developed by Gendreau et al. (2001). They showed that their method, called Divide and Merge Heuristic performed better than Tabu Search.

Kurz and Askın (2001) also compared the results of different heuristics including Slicing Heuristic, Multiple MULTI-FIT Heuristic, Multiple Insertion Heuristic, and Genetic Algorithm for the problem of parallel line scheduling times where ready times were allowed. They defined ready time as “a time before which the job may not begin processing or, if set-ups are required, a time before which a set-up time for the job may not begin”. They used 100 jobs to be performed on 10 machines for comparison. The results showed that the performance of Genetic Algorithms was impressive for some problems while Multiple Insertion was the best for all over performance.

Kim et al. (2003) proposed a 4 staged heuristics for the problem of parallel line scheduling based on due date density categorization. They listed jobs, categorized according to due date density, applied tabu search for better sequencing and allocated jobs to machines. They argued that their method performed 6-28% better than Ovacik and Uzsoy’s (1995) Rolling Horizon Procedure (RHP). They also analyzed that when the number of jobs increased Total Weighted Tardiness increased for both RHP and newly developed method. Similarly, as the number of machines increased, total weighted tardiness decreased for RHP and newly developed method.

Cakici and Mason (2007) worked on the process of manufacturing of integrated circuits on silicon wafers in order to solve parallel machine scheduling problem subjecting auxiliary resource constraints. The auxiliary resource in their study was reticle availability. Integrated circuits were built by adding ‘layers’ with desired properties on the silicon wafer’s surface. Every layer of each product can require its own unique reticle. Cakici and Mason stated that similar kind of auxiliary resources existed in many manufacturing systems. However, there was a dearth of research that considered auxiliary resource constraints in scheduling. They developed two heuristics based on two existing heuristics and proved that the one with post processing local search procedures performed better.

In recent years, Constraint Programming (CP) has been used as an alternative to Integer Programming (IP) and Mixed Integer Programming (MIP) in scheduling problems. Edis and Ozkarahan (2011) used this technique on a resource constraint identical parallel machine scheduling problem with machine eligibility restrictions. In this study, they not only considered machines but also additional resources like vehicles, machine operators, tools, pallets, industrial robots and so on. Furthermore, they concentrated on machine eligibility, i.e., some jobs had to be processed on specific machines. They benefited from three optimization models, Integer Programming, Constraint Programming and combination of both. They stated that CP was a suitable technique for sequencing and scheduling applications, strict feasibility problems and highly constrained problem, which was in line with the prior studies (Smith et al. 1997, Darbi-Dowman and Little 1998, Jain and Grossmann 2001, Lustig and Puget 2001). However, CP also had weaknesses and therefore it should had to be combined with other methods. The results indicated that IP/CP combination outperformed individual IP and CP models in almost all of 200 test problems.

Edis and Oğuz (2012) handled Parallel Machine Flexible Resources Scheduling (PMFRS) problem. In this type of system, processing time could be decreased by additional resources. PMFRS has two types: dynamic and static. In static version, freely allocated resources cannot be changed during scheduling horizon while in dynamic version resources can be allocated on any time throughout scheduling horizon. In this study, Edis and Oğuz (2012) extended the IP model of the dynamic PMFRS problem developed by Daniels et al. (1996) and combined the relaxed IP model of Grigoriev et al. (2005) with a CP model. Then, compared the results of all methods.

Lara et al. (2012) worked on the minimization of total tardiness of identical parallel machines scheduling problem. They used an adapted version of Biskup et al. (2008)'s model and applied local and tabu search algorithms. The results indicated that tabu search performed better for large set of parallel machines compared to local search algorithm and ABHG when jobs do not have common due dates. ABHG is a heuristic technique which is another version of Biskup et al. (2008) developed by Yalaoui (2012).

Gedik et al. (2016) provided one of the most recent studies where they developed a CP model and logic-based Benders algorithm for non-identical parallel line scheduling with sequence dependent setup times and job availability intervals. Job availability means that jobs are available only in some intervals. Furthermore, there were budget limitations on the total costs. They argued that CP model found feasible solutions but could not provide optimal solutions in a limited time interval. In order to overcome this problem, they applied a logic-based Benders decomposition technique and obtained better results.

As we try to show, there are numerous studies about scheduling problems even if we add some filtering criteria. Still, new papers appear every year. This condition stems from two main reasons. The first one is combinatorial hardness of the problem. Especially for large size problems, it is very difficult to reach optimal solution in a short time or it is not possible to find a good solution. Therefore, researchers work on generating faster methods. The second and more critical reason is that theoretical findings about scheduling do not meet the practical needs because scheduling is related with other processes like Material Requirement Planning, Production and Inventory Planning, Sales Planning, etc. With this motivation, in this study Production Planning and Scheduling problems are handled together.

2.2 Production planning and scheduling

Planning and scheduling has a special importance to implement effective supply chain management. But its implementation would not be easy with the conventional information systems. In traditional production systems, production planning and scheduling are done sequentially. Someone determine the plan then scheduling is performed. In those types of systems, response time of scheduling to unexpected changes in the production phase is very poor. Therefore, practitioners seek ways to handle flexible production planning and scheduling system. Studies similar to the current thesis study are as follows:

Hastings et al. (1982) did one of the first studies which integrated scheduling with production planning. They realized that realistic schedule was possible if it was well coordinated with material requirement processes. Therefore, they proposed an integrated approach to scheduling and Material Requirement Planning (MRP). Lasserre (1992) argued that top-down hierarchical integrated systems created unfeasible schedules. Hence, he developed a decomposition approach which alternate between production plan problem with a fixed choice of a sequence of products and a scheduling problem for a fixed choice of the production plan. He aimed that there should exist at least one feasible schedule for every feasible plan.

A pioneering study was done by Özdamar et al. (1998). They developed a Hierarchical Decision Support System (HDSS) for advanced production planning which included aggregate planning, family and item planning. They argued that Decision Support Systems had great significance because it enabled planners to easily make feasible plans without any mathematical background. Their application provided opportunity to combine any modeling options by using verbal expression including overtime production capacity, subcontracting production capacity, backorders, hiring and firing of workforce, safety stocks. They used their interactive HDSS for a real factory manufacturing agricultural engines successfully.

Chen and Ji (2007) proposed a mixed integer model for an integrated production planning and scheduling system. Their model considers capacity constraint, operations sequences lead time and due dates. The objective of the model was minimizing the production time, earliness and tardiness. A number of different size problems were solved using software CPLEX on a Personal Computer. They stated that the established method found optimal results. But since the problem was NP-hard meta-heuristics could also be used.

Erdirik-Dogan and Grossman (2007) studied on planning models for parallel batch reactors with sequence-dependent changeovers. The first one was Relaxed Planning (RP) model; this model did not consider exact changeover times. The objective of the model was to determine the optimal production quantities for each item on a time horizon. The second one was Detailed Planning (DP) model, with additive constraints that was used to determine the optimum sequence of products. The first model could be easily solved in short time periods using Branch and Bound enumeration periods, however, detailed model required a long execution time. Therefore, a heuristic forward Rolling Horizon (RH) algorithm was applied to detailed model. This method decomposed the problem into sub problems. Each sub problem solved with DM while other parts solved with RP. For small sized problems, RH and DP reached similar results whereas for larger problems RH underestimated the profit by up to 10%.

Omar and Tao (2007) proposed a three level Hierarchical Production Planning (HPP) and scheduling approach for multi-product identical parallel machines in a batch process environment. They argued that production planning and scheduling in HPP context was extended with their study in three ways. First, they considered setups at the aggregate level. Second, an optimal solution approach was developed for difficult batching of orders phase. Third, sequencing problem on parallel machines with distinct due date was solved. They implemented their theoretical model with actual factory values and concluded that their approach provided cost saving by nearly 9.6%.

An optimization tool was developed by Josefowska and Zimniak (2008). They defined their subject problem as production planning and scheduling on unrelated parallel machines with sequence dependent setup times. For the solution of multi-objective optimization problem a genetic algorithm was developed and a new mechanism was added to reduce the search space of the algorithm by removing solutions that were evidently dominated by other solutions. The concept was based on an observation that a domain expert could provide partial knowledge that was not sufficient to construct a final solution but could be used to reduce the search space.

(Josefowska and Zimniak, 2008). They argued that missing the optimal result was possible but the intuition of an expert was very accurate in most cases. Experiments done by Josefowska and Zimniak (2008) proved that the reduction of the search space reduced computational time of the algorithm with little impact on the quality of solutions. After genetic algorithm was applied, the best solution was chosen by the planner from the set of potentially Pareto-optimal solutions. They implemented their system to a plastic pipe plant in Poland and integrated with the ERP system Microsoft Axapta used in the factory.

Lawryowicz (2008) developed a model by considering issues: how to make flexible plans, how to make efficient scheduling considering dynamic situations and how to make an integrated system including various constraints. His model had two parts: the first part was a traditional approach including genetic algorithm which created a preliminary and possibly suboptimal solution. The second part was use of a combined method of expert system and genetic algorithm on the first part solution. The second part produce a more detailed production plan based on customer demand and manufacturing reports. This method was applied to 10 different real factories with different parameters. The schedules were compared with the results of combination of different dispatching rules (Earliest Due Date, Least Operations Remaining, Shortest Processing Time etc.). The results indicated that genetic algorithm created schedules with makespan that was on average 21% better than the methods based on dispatching rules.

The most inspiring study for this thesis was by Xue et al. (2011). They developed a Hierarchical Production Planning (HPP) approach for the problem of integrated production planning with sequence dependent family setups. They used the model developed by Ozdamar et al. (1998) which considered Aggregate

Production Planning (APP) before item disaggregation and item scheduling. The APP model determined production, inventory, subcontracting and backordering levels, regular time and overtime, and labor employment. After APP, family disaggregation was applied by aiming minimum setup costs between family types. They applied newly developed integrated method and HPP with different inventory costs and seasonality and concluded that total cost of integrated model was consistently less than HPP. They also stated that their assumption that all families were produced in each period should be relaxed.

The current thesis study is similar to Xue et al. (2011) since both studies consider integrated production planning scheduling problem with sequence dependent setup times. However they differ in a basic model assumption: For the sake of formulation simplicity, in the previous study it was assumed that every product had to be produced every day whereas this assumption is relaxed in the thesis study. This results in a much more complex formulation with the added binary variables and nonlinear constraints; however in the thesis, the original quadratic model is further improved to linearize the nonlinear constraints. The second assumption of the previous study was that the demand was always satisfied. However, in our model we consider the case which all demands may not be satisfied because of capacity constraints, machine breakdowns or other unexpected events. This model handles those type of problems by considering lost sale cost of each product when we have to choose not meet the demands of some products. We believe that our model more realistic because it is very common to have issues which reduce the capacity. Besides, their model has more tendency to get infeasible results due to unsatisfied demand issues. We should also note that the previous model consider labor employment and dismissal plans and overtime capacity issues. Our model does not consider those conditions because of the fact that in our subject factory overtime and hiring firing is not possible.

We can see that there are studies in the literature which handle production planning and scheduling problems together; however the number of such integrated studies is few. At this point, the current study is unique because of the following main reasons:

The first point is that we develop a more general and valid integrated model where unreasonable restrictions of the previous studies are relaxed. We believe that the current integrated model is more flexible than the existing studies and can be successfully implemented in a wider range of manufacturing systems.

Secondly, the generated model is a linear model which has less computational complexity than a nonlinear model. This simplification allows us not to search for heuristics methods, at least for the solution of moderate size problems. Scheduling problems are generally formulated as MILP, as in our problem. In most cases, the formulation leads to an NP-Hard problem and heuristics methods are very widely used. The strength of heuristics methods is that they may achieve a solution in a short span of time. However, they cannot guarantee optimal solution. In this regard, it is very risky to use those type of studies in real life. In our application, we used solver BARON that uses branch and bound scheme to solve integer programming models with moderate size of variables.

The third contribution is Decision Support System (DSS). DSS makes a study more meaningful in terms of practical use. Our model is ready to use for any parallel line production environment without any extra effort. Ozdamar et al. (1998) is the only study that and has a DSS for production planning and scheduling. Their work was inspirational for us but their model was hierarchical and their desktop application needed user manipulation between stages. Our DSS only needed master data about products, lines and related costs.

CHAPTER 3

PROBLEM DEFINITION AND MODEL DEVELOPMENT

In this study, we consider the continuous manufacturing system of leading detergent production brand in Turkey. The production process includes two stages: The first stage is producing the semi-finished product and second stage is the packaging of the semi-finished products with some additive materials in different sizes to form the finished products.

Let us note that the word powder will be used for semi-finished products. Actually, it is the common base material of detergents with similar ingredients. And the words finished product and detergent will be used interchangeably.

3.1 About detergent production

In our subject factory, the powder is being produced in a machine called Postdoze. Postdoze is a special kind of blender in which the ingredients of powder are mixed. Then, this powder can be either directly fed into packing lines to produce different types of detergents or moved to stocks. As a company policy, the factory does not hold finished product stocks but they keep powder stocks for a few weeks.

In this powder production system, backordering is not allowed. If the available powder on hand is not enough to satisfy demanded quantity, powder can be subcontracted within certain limits. For conditions which subcontracting is also not enough, production planners are obligated not to meet the demands of some detergents. At this stage, production planners choose not to produce the finished products (and therefore powders) with the lowest lost sale costs.

The demand and order processes are as follows: Since the company does not hold finished product stocks, demand data has a critical role in the production quantity decisions. At the beginning of each week, marketing department provides the actual detergent demands of the current week and forecasts for the demands of the coming weeks. Then, weekly production and inventory control plans are made for the powders based on 8 weeks rolling horizon policy. After that, weekly production schedules are generated for the finished products, i.e., detergents. However this schedule is subject to several updates during its implementation, which urges the planners to reschedule the production of detergents on the bands. These updates are necessary due to unexpected detergent demand changes, shortage of any raw material or powder, or due to machine breakdowns.

This detergent factory produces 21 different powder types and each powder type is used to produce 1-5 different detergent products that vary in their sizes and/or perfumes they include. In total there are 70 different detergents being produced on 7 identical packing lines. In this complexity, scheduling the production of detergent types in the packing lines is a critical issue because changing detergent types or powder types may take a long setup time and this incurs a cost. In the packing lines, there are two basic types of production changeover, namely, size change and powder change. Size change refers to start producing a detergent with a different size, i.e., same powder is to be packed in a different size. For the size change process, some settings of the machines on the line should be updated and packing materials should be changed. On the other hand, powder change refers to start using a different powder in the packing line. For the powder change process, the pipes on the lines should be cleaned by using special solvents in order to prevent mixing different types of powders during packing process. Sometimes both size and powder change occurs at the same time and those setup processes are more costly and time consuming.

The production capacity is yet another issue that affects production planning and scheduling. The factory works 7 days per week and 24 hours in a day, in three 8hr. shifts. Noting that the number of labor and machines are not frequently changing, we will assume that available working hours per day or per week are fixed.

3.2 Problem formulation

In this section, we develop an integrated mathematical model to optimize the production and scheduling of detergent production process so that the relevant costs are minimized. The model has two main parts: The first part is about production and inventory planning of powder. The quantity of powders to be produced, inventoried and subcontracted is calculated according to demand data and inventory level on hand.

The second part of the model is the scheduling of finished products on the parallel and identical production lines. The model seeks to find the best production sequence of detergents on each parallel and identical line. This production sequence significantly affects the total costs since the setup costs are sequence dependent. We further note that setup and process times are deterministic in our model.

A basic assumption in our formulation is on the maximum number of jobs that can be assigned into a production line in a day. Practically, factory produces a maximum of 4 different products in each line in a day. Hence without loss of generality in our model, a maximum of 4 job positions are allowed in each line for each day. Noting that in the current capacity setting the factory works for 24 hours a day, fixing the maximum number of jobs in a line to 4 is not a very strict assumption.

Moreover this assumption simplifies the formulation in terms of the number of constraints and variables.

The basic aim of the model is the minimization of total cost of powder and detergent productions processes.

In the next section, a mixed integer quadratic programming, MIQP model is generated to optimize the integrated production and scheduling problem for detergent production. The performance of the model is tested in problems with different sizes. Later, in section 3.3.2, MIQP model is further improved by linearizing the nonlinear equations, and a mixed integer linear programming, MILP model is obtained. The performance of the improved model and the original model are compared in problems with different sizes.

3.2.1 Model 1: Mixed Integer Quadratic Programming (MIQP) model

The problem formulation of the MIQP Model is as follows:

Indices

- i: Powder type, i=1,2,3,…,I
- j: Packing size for powder i, j=1,2,..,J_i. Hence (i, j) specifies an end product type, i.e., a detergent with powder i and packing size j.
- k: Packing line, k=1,2,…,K
- t: Day of a week, t=1,2,…,T
- w: Week index, w=1,2,…,W
- p: Production order index in a line in a day, p=1,2,3,4
- s: Number of types of products assigned to a line in a day, s=0,1,2,3,4

Parameters

- d_{ijw}: Demand for product (i, j) in week w.

- $f_{iji'j'k}$: Setup duration required to change over from product (i, j) to (i', j') in line *k*.
- $size_{ij}$: Amount of powder *i* in product (i, j).
- $duration_{ijk}$: The production time of product *(i, j)* in line *k*.
- c_i^{inv}: Unit inventory holding cost for powder *i* per day.
- c_i^{sub}: Unit subcontraction cost for powder *i*.
- c_{ij}^{lost}: Unit lost sales cost for product *(i, j).*
- c_{ijk}^{pack}: Unit packing cost of product *(i, j)* in line *k*.
- c_i^{pow}: Unit production cost of powder *i* in post doze.
- $c_{iji'j'k}^{setup}$: Setup cost of shifting from product (*i*, *j*) to (*i'*, *j'*) in line *k*.
- $dailyLineCap_k$: Daily working time capacity of line *k*.
- $dailyPostCap$: Daily working time capacity of postdoze.

Decision Variables

- I_{itw}: Amount of inventory kept for powder *i* in day *t* of week *w*.
- S_{itw}: Amount of subcontracting for powder *i* in day *t* of week *w*.
- X_{itw}: Amount of powder *i* produced in post doze in day *t* of week *w*.
- Q_{ijktw}: Number of packages of product type (*i*, *j*) produced in line *k* on day *t* of week *w*.
- π_{sktw} : 1, if the number of types of products to be produced in line *k* in day *t* of week *w* is *s*, and 0 otherwise.
- A_{ktw}: The total setup time spent in line *k* in day *t* of week *w*.
- μ_{ijktw}: 1, if the quantity $Q_{ijktw}$is greater than 0, and 0 otherwise.
- η_{ijpktw}: 1, if the quantity $Q_{ijktw}$is produced at the *p*th position, and 0 otherwise.

- $Y_{iji'j'pktw}$: 1, if product (*i*, *j*) is produced at *p*th position and product (*i*′, *j*′) is produced at (*p*+*1*)th position, in line *k* in day *t* of week *w*, and 0 otherwise.

Constraints (1-5)

$$\sum_{k=1}^{K}\sum_{j=1}^{j_i} Q_{ijktw} * size_{ij} = X_{itw} + I_{i(t-1)w} - I_{itw} + S_{itw} \qquad \forall\, i,t,w \qquad (1)$$

$$I_{i7w} = I_{i1(w+1)} \qquad \forall\, i,t,w \qquad (2)$$

$$\sum_{t=1}^{T}\sum_{k=1}^{K} Q_{ijktw} \leq d_{ijw} \qquad \forall\, i,j,w \qquad (3)$$

$$\sum_{i=1}^{I}\sum_{j=1}^{j_i} Q_{ijktw} * duration_{ijk} + \mathrm{A}_{\mathrm{ktw}} \leq dailyLineCap_k \qquad \forall\, k,t,w \qquad (4)$$

$$\sum_{i=1}^{I} X_{itw} \leq dailyPostCap \qquad \forall\, t,w \qquad (5)$$

Constraint 6

$$\sum_{i=1}^{I}\sum_{j=1}^{j_i}\sum_{i'\neq i}^{I}\sum_{j'\neq j}^{j_i} f_{iji'j'k} * \sum_{p=1}^{4} Y_{iji'j'pktw} = A_{ktw} \qquad \forall\, k,t,w \qquad (6)$$

Constraint (7-8)

$$\mu_{ijktw} \leq Q_{ijktw} \qquad \forall\, i,j\ k,t,w \qquad (7)$$

$$\mu_{ijktw} * Q_{ijktw} \geq Q_{ijktw} \qquad \forall\, i,j\ k,t,w \qquad (8)$$

Constraint 9

$$\pi_{0ktw} + \pi_{1ktw} + \pi_{2ktw} + \pi_{3ktw} + \pi_{4ktw} = 1 \qquad \forall\, k,t,w \qquad (9)$$

Constraint (10-13)

$$\pi_{1ktw} * \eta_{ij1ktw} = \mu_{ijktw} * \pi_{1ktw} \qquad \forall\, i,j,k,t,w \qquad (10)$$

$$\pi_{2ktw} * (\eta_{ij1ktw} + \eta_{ij2ktw}) = \mu_{ijktw} * \pi_{2ktw} \qquad \forall\, i,j,k,t,w \qquad (11)$$

$$\pi_{3ktw} * (\eta_{ij1ktw} + \eta_{ij2ktw} + \eta_{ij3ktw}) = \mu_{ijktw} * \pi_{3ktw} \qquad \forall\, i,j,k,t,w \qquad (12)$$

$$\pi_{4ktw} *(\eta_{ij1ktw} + \eta_{ij2ktw} + \eta_{ij3ktw} + \eta_{ij4ktw}) = \mu_{ijktw} * \pi_{4ktw} \qquad \forall\, i,j,k,t,w \quad (13)$$

Constraints (14-17)

$$\sum_{i=1}^{I} \sum_{j=1}^{j_i} \eta_{ij1ktw} = 1 - \pi_{0ktw} \qquad \forall\, k,t,w \quad (14)$$

$$\sum_{i=1}^{I} \sum_{j=1}^{j_i} \eta_{ij2ktw} = 1 - \pi_{0ktw} - \pi_{1ktw} \qquad \forall\, k,t,w \quad (15)$$

$$\sum_{i=1}^{I} \sum_{j=1}^{j_i} \eta_{ij3ktw} = 1 - \pi_{0ktw} - \pi_{1ktw} - \pi_{2ktw} \qquad \forall\, k,t,w \quad (16)$$

$$\sum_{i=1}^{I} \sum_{j=1}^{j_i} \eta_{ij4ktw} = 1 - \pi_{0ktw} - \pi_{1ktw} - \pi_{2ktw} - \pi_{3ktw} \qquad \forall\, k,t,w \quad (17)$$

Constraint 18

$$\eta_{ijpktw} * \eta_{i'j'(p+1)ktw} = Y_{iji'j'pktw} \qquad \forall\; p,k,t,w, \quad (18)$$

Objective Function

Minimize Total Cost=

$$\sum_{i=1}^{I} \sum_{t=1}^{T} \sum_{w=1}^{W} \left[c_i^{sub} * S_{itw} + \mathrm{c}_{\mathrm{i}}^{\mathrm{inv}} * I_{itw} + c_i^{pow} * X_{itw} + \sum_{j=1}^{j_i} \sum_{k=1}^{K} c_{ijk}^{pack} * Q_{ijktw} \right] + \sum_{i=1}^{I} \sum_{j=1}^{j_i} \sum_{w=1}^{W} c_{ij}^{lost} * \left(d_{ijw} - \sum_{t=1}^{T} \sum_{k=1}^{K} Q_{ijktw} \right) + \sum_{w=1}^{W} \sum_{t=1}^{T} \sum_{k=1}^{K} \sum_{p=1}^{P} \sum_{i=1}^{I} \sum_{j=1}^{j_i} \sum_{i'=1}^{I} \sum_{j'=1}^{j_{i'}} c_{iji'j'k}^{setup} * Y_{iji'j'pktw} \quad (19)$$

Q_{ijktw} is nonnegative integer variable.

$S_{itw}, I_{itw}, X_{itw}, A_{ktw}$ are nonnegative continuous variables.

$\pi_{sktw}, \mu_{ijktw}, \eta_{ijpktw}, Y_{iji'j'pktw}$ are binary variables.

3.2.1.1 Explanations of the constraints

Constraints 1-5 are standard restrictions for production and inventory planning of powders.

1. The first constraint is the Inventory Balance Equation for powder types. It is obtained by balancing production, inventory and subcontraction quantities in day *t* of week *w*.

2. Inventory holding constraint between two weeks.

3. Demand constraint for final products: The number of packages of product *(i, j)* produced all lines in a week should not exceed the weekly demand for that product.

4. Production Capacity Constraint: Total daily workload in a line should not exceed the daily line capacity for that line, i.e., daily production time plus total daily setup time in line *k* should not exceed the daily line capacity for line *k* for every day *t* of week *w*.

5. Postdoze Production Capacity Constraint: Total daily workload in postdoze should not exceed the daily capacity of the machine.

Constraints 6-18 are restrictions for scheduling detergent productions on parallel and identical production lines.

6. By the initial assumption, there can be at most 4 available production positions in a production line at any day. π_{sktw} denotes the number of production positions which are allocated to the production of any type of product in line *k* at any day. For example if π_{3ktw} is 1, then there are 3 production positions allocated for production in line *k*, leaving the last position unassigned. Similarly, if π_{0ktw} is 1, then there is

no production at line *k* in that day of week. Apparently, there is either 0,1,2,3 or 4 production positions assigned for production at line *k* in any day.

If a product *(i, j)* is produced, i. e, Q_{ijktw} > 0, it is assigned to a single production position in a production line.

7. The total setup time in line k in a certain day is calculated by the sum of setup durations that take place in that line in that day. $Y_{iji'j'pktw}$ is a binary variable indicating if there is a setup between product (i,j) and (i',j') at position p in any line k in a day. Multiplying these with the setup durations, $f_{iji'j'k}$ adding over the positions p provides the total setup duration in line *k* in a day.

8-9. If product (i, j) is produced in line *k* in day *t* of week *w*, i.e., Q_{ijktw}> 0, then μ_{ijktw} = 1. Otherwise, μ_{ijktw} = 0

10-13. By equations 6-7, if product *(i, j)* is produced in line *k* in day *t* of week *w*, then $\mu_{ijktw} = 1$. By equation 4, there are s available positions for its production, indicated by π_{sktw}=1. Equations 8-11 assure that product *(i, j)* should be produced in one of these s positions. For example, if π_{2ktw}=1, then there are 2 positions available for production at line k in that day. In this case equation 9 will be a valid constraint, whereas equation 8, 10, 11 will be redundant constraints. Equation 9 will be used to assign the production of *(i, j)* to either position 1 or position 2 at line *k* in that day, letting either η_{ij1ktw} or η_{ij2ktw} equal to 1.

14-17. Equations 8-11 assign the production of *(i, j)* to any of the available *s* positions at line *k* in a day. Equations 12-15 assure that no two products are assigned to the same position. Moreover if *s* positions are available then there should be no production in the last *4-s* positions. For example, if there are 2 positions available at line *k* at any day, i.e., π_{2ktw}=1, then there will a single production assignment to position 1 and position 2 respectively by equations 12 and 13; whereas there will be no production assignment to positions 3 and 4 by equations 14 and 15.

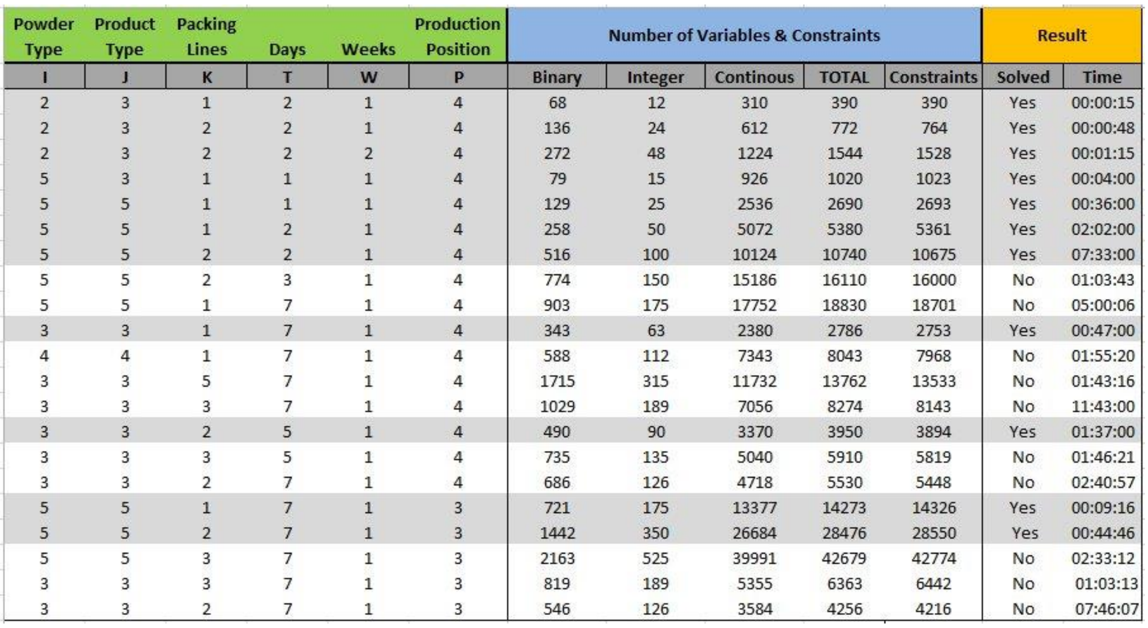
18. If product *(i, j)* is produced at position p and product *(i', j')* is produced at position *(p+1)* in line *k* in a day, then $Y_{iji'j'pktw}$=1. Otherwise it is zero.

Objective Function to be minimized is given in equation 17, calculated as the sum of Subcontraction Cost, Inventory Holding Cost, Powder Production Cost, Finished Product Packing Cost, Lost Sale Cost and Setup Cost.

3.2.1.2 Performance of MIQP model

Our first model is a MIQP model which is nonlinear due to the quadratic terms in equations 8, 10, 11, 12, 13 and 18. In this section we test our model by using the solver BARON and show that although it performs well for very small size problems, solution for moderate or large size problems is either impossible or requires a long execution time. Performance of MIQP model is shown below in Table 1.

Table 1. The Performance of MIQP model



Powder Type	Product Type	Packing Lines	Days	Weeks	Production Position	Number of Variables & Constraints					Result	
I	J	K	T	W	P	Binary	Integer	Continous	TOTAL	Constraints	Solved	Time
2	3	1	2	1	4	68	12	310	390	390	Yes	00:00:15
2	3	2	2	1	4	136	24	612	772	764	Yes	00:00:48
2	3	2	2	2	4	272	48	1224	1544	1528	Yes	00:01:15
5	3	1	1	1	4	79	15	926	1020	1023	Yes	00:04:00
5	5	1	1	1	4	129	25	2536	2690	2693	Yes	00:36:00
5	5	1	2	1	4	258	50	5072	5380	5361	Yes	02:02:00
5	5	2	2	1	4	516	100	10124	10740	10675	Yes	07:33:00
5	5	2	3	1	4	774	150	15186	16110	16000	No	01:03:43
5	5	1	7	1	4	903	175	17752	18830	18701	No	05:00:06
3	3	1	7	1	4	343	63	2380	2786	2753	Yes	00:47:00
4	4	1	7	1	4	588	112	7343	8043	7968	No	01:55:20
3	3	5	7	1	4	1715	315	11732	13762	13533	No	01:43:16
3	3	3	7	1	4	1029	189	7056	8274	8143	No	11:43:00
3	3	2	5	1	4	490	90	3370	3950	3894	Yes	01:37:00
3	3	3	5	1	4	735	135	5040	5910	5819	No	01:46:21
3	3	2	7	1	4	686	126	4718	5530	5448	No	02:40:57
5	5	1	7	1	3	721	175	13377	14273	14326	Yes	00:09:16
5	5	2	7	1	3	1442	350	26684	28476	28550	Yes	00:44:46
5	5	3	7	1	3	2163	525	39991	42679	42774	No	02:33:12
3	3	3	7	1	3	819	189	5355	6363	6442	No	01:03:13
3	3	2	7	1	3	546	126	3584	4256	4216	No	07:46:07

In Table 1, the 'Solved' column of Result Tab shows the success of MIQP model. As it is can be seen the model can be used to solve a maximum of 25 different products, i.e., I=5, J=5, with a maximum of two production lines and for a one week plan. In this setting the maximum number of decision variables is over 28,000 and the number of constraints is over 26,000. The problem becomes unsolvable or a very long execution time is needed for larger size problems. This condition makes our model inappropriate for the practical needs, since in the current system there are 70 different products, 7 production lines and 8 weeks. Therefore, an enhancement in the current formulation is needed.

3.2.2 Improving MIQP model by linearizing constraints

In the previous section we observe that the MIQP model has a poor performance in obtaining the optimal solution in moderate or large scale problems. In this section we convert the MIQP model into an equivalent linear model by linearizing the nonlinear constraints. Moreover, we convert the integer decision variable Q_{ijktw} to a continuous varaible to decrease the complexity of the resulting model.

Noting that the variables in the nonlinear equations are all binary, by using the techniques in (Watters, 1967) we make the following modifications:

3.2.2.1 Linearizing constraint 8:

Constraint 8 in equation 8 indicates that when $Q_{ijktw} > 0$, then μ_{ijktw}=1; otherwise μ_{ijktw}=0. This restriction can be provided in a linear form below, as in equation 8A.

$$\mu_{ijktw} * d_{ijw} \geq Q_{ijktw} \qquad \forall i, j\ k, t, w \qquad (8A)$$

Noting that $d_{ijw} \geq Q_{ijktw}$ by equation 2, $Q_{ijktw} > 0$ implies that $d_{ijw} > 0$. Hence when $Q_{ijktw} > 0$, then μ_{ijktw}=1.

3.2.2.2 Linearizing constraints 10-13:

Constraint 10 in equation 13 can be reconstructed by replacing it with the pair of equations 9A and 9B to obtain a linear set of constraints that provide the same output.

$$-\eta_{ij1ktw} + \mu_{ijktw} \leq 1 - \pi_{1ktw} \qquad \forall\, i,j,k,t,w \quad \text{(10A)}$$

$$\eta_{ij1ktw} - \mu_{ijktw} \leq 1 - \pi_{1ktw} \qquad \forall\, i,j,k,t,w \quad \text{(10B)}$$

In equation 10, if $\pi_{1ktw} = 1$, then we have $\eta_{ij1ktw} = \mu_{ijktw}$. Similarly if $\pi_{1ktw} = 1$ in equations 10A and 10B, we have $\mu_{ijktw} \leq \eta_{ij1ktw}$ and $\eta_{ij1ktw} \leq \mu_{ijktw}$, resulting in $\eta_{ij1ktw} = \mu_{ijktw}$.

In equation 10, if $\pi_{1ktw} = 0$, then the equation is redundant. Similarly if $\pi_{1ktw} = 0$ in equations 10A and 10B, we have $\mu_{ijktw} \leq 1 + \eta_{ij1ktw}$ and $\eta_{ij1ktw} \leq 1 + \mu_{ijktw}$, resulting in no restrictions on η_{ij1ktw} and μ_{ijktw}.

Hence we see that Constraint 10 in equation 10 can be replaced by the constraints in equations 10A and 10B.

Using the same technique we linearize the equations 11-13 by replacing them with pairs of linear constraints in equations 11A, 11B, 12A, 12B, 13A, 13B.

$$\eta_{ij1ktw} + \eta_{ij2ktw} - \mu_{ijktw} \leq 2(1 - \pi_{2ktw}) \qquad \forall\, i,j,k,t,w \quad \text{(11A)}$$

$$-(\eta_{ij1ktw} + \eta_{ij2ktw}) + \mu_{ijktw} \leq 2(1 - \pi_{2ktw}) \qquad \forall\, i,j,k,t,w \quad \text{(11B)}$$

$$(\eta_{ij1ktw} + \eta_{ij2ktw} + \eta_{ij3ktw}) - \mu_{ijktw} \leq 3(1 - \pi_{3ktw}) \qquad \forall\, i,j,k,t,w \quad \text{(12A)}$$

$$-(\eta_{ij1ktw} + \eta_{ij2ktw} + \eta_{ij3ktw}) + \mu_{ijktw} \leq 3(1 - \pi_{3ktw}) \qquad \forall\, i,j,k,t,w \quad \text{(12B)}$$

$$(\eta_{ij1ktw} + \eta_{ij2ktw} + \eta_{ij3ktw} + \eta_{ij4ktw}) - \mu_{ijktw} \leq 4(1 - \pi_{4ktw}) \qquad \forall\, i,j,k,t,w \quad \text{(13A)}$$

$$-(\eta_{ij1ktw} + \eta_{ij2ktw} + \eta_{ij3ktw} + \eta_{ij4ktw}) + \mu_{ijktw} \leq 4(1 - \pi_{4ktw}) \qquad \forall\, i,j,k,t,w \quad \text{(13B)}$$

3.2.2.3 Linearizing constraint 18:

In Constraint 18 shown in equation 18, $Y_{iji'j'pktw}$ is binary variable expressed as the multiplication of two binary variables η_{ijpktw} and $\eta_{i'j'(p+1)ktw}$. This can be written in a linear form as given below in equation 18A.

$$Y_{iji'j'pktw} >= \eta_{ijpktw} + \eta_{i'j'(p+1)ktw} - 1 \qquad \forall\ p,k,t,w,i,j,i',j' \quad (18A)$$

Since $Y_{iji'j'pktw}$ is a cost term in the objective function to be minimized in equation 18, in the optimal solution it will be 1 if only if both η_{ijpktw} and $\eta_{i'j'(p+1)ktw}$ are 1. Hence constraint 18 can be replaced by equation 18A without affecting its outputs.

Linearizing the integer decision variable Q_{ijktw}

Without loss of generality we can convert the integer variable Q_{ijktw} to a continuous decision variable. This decreases the number of integer variables involved in the MILP model resulting in a less complex model formulation. The resulting MILP model can be restated as follows:

3.2.3 Model 2: Mixed Integer Linear Programming (MILP) model

Constraints (1-4)

$$\sum_{k=1}^{K}\sum_{j=1}^{j_i} Q_{ijktw} * size_{ij} = \mathrm{X_{itw}} + \mathrm{I_{it(w-1)}} - \mathrm{I_{itw}} + \mathrm{S_{itw}} \qquad \forall\, i,t,w \qquad (1)$$

$$I_{i7w} = I_{i1(w+1)} \qquad \forall\, i,t,w \qquad (2)$$

$$\sum_{t=1}^{T}\sum_{k=1}^{K} Q_{ijktw} \le d_{ijw} \qquad \forall\, i,j,w \qquad (3)$$

$$\sum_{i=1}^{I}\sum_{j=1}^{j_i} Q_{ijktw} * duration_{ijk} + \mathrm{A_{ktw}} \le DailyLineCap_k \qquad \forall\, k,t,w \qquad (4)$$

$$\sum_{i=1}^{I} X_{itw} \le dailyPostCap \qquad \forall\, t,w \qquad (5)$$

Constraint 6

$$\sum_{\mathrm{i=1}}^{I} \sum_{j=1}^{j_i} \sum_{i'\neq i}^{I} \sum_{j'\neq j}^{j_i} f_{iji'j'k} * \sum_{p=1}^{4} Y_{iji'j'pktw} \quad = A_{ktw} \qquad \forall\, k, t, w \qquad (6)$$

Constraint (7-8A)

$$\mu_{ijktw} \ \leq \ Q_{ijktw} \qquad \forall i, j\ k, t, w \quad (7)$$

$$\mu_{ijktw} * d_{ijw} \ \geq Q_{ijktw} \qquad \forall i, j\ k, t, w \quad (8A)$$

Constraint 9

$$\pi_{0ktw} + \pi_{1ktw} + \pi_{2ktw} + \pi_{3ktw} + \pi_{4ktw} = 1 \qquad \forall\, k, t, w \qquad (9)$$

Constraints (10A, B – 13A, B)

$$-\eta_{ij1ktw} + \mu_{ijktw} \leq 1 - \pi_{1ktw} \qquad \forall\, i, j, k, t, w \quad (10A)$$

$$\eta_{ij1ktw} - \mu_{ijktw} \leq 1 - \pi_{1ktw} \qquad \forall\, i, j, k, t, w \quad (10B)$$

$$\eta_{ij1ktw} + \eta_{ij2ktw} - \mu_{ijktw} \ \leq 2(1 - \pi_{2ktw}) \qquad \forall\, i, j, k, t, w \quad (11A)$$

$$-(\eta_{ij1ktw} + \eta_{ij2ktw}) + \mu_{ijktw} \ \leq 2(1 - \pi_{2ktw}) \qquad \forall\, i, j, k, t, w \quad (11B)$$

$$\eta_{ij1ktw} + \eta_{ij2ktw} + \eta_{ij3ktw} - \mu_{ijktw} \ \leq 3(1 - \pi_{3ktw}) \qquad \forall\, i, j, k, t, w \quad (12A)$$

$$-(\eta_{ij1ktw} + \eta_{ij2ktw} + \eta_{ij3ktw}) + \mu_{ijktw} \ \leq 3(1 - \pi_{3ktw}) \qquad \forall\, i, j, k, t, w \quad (12B)$$

$$\eta_{ij1ktw} + \eta_{ij2ktw} + \eta_{ij3ktw} + \eta_{ij4ktw} - \mu_{ijktw} \ \leq 4(1 - \pi_{4ktw}) \qquad \forall\, i, j, k, t, w \quad (13A)$$

$$-(\eta_{ij1ktw} + \eta_{ij2ktw} + \eta_{ij3ktw} + \eta_{ij4ktw}) + \mu_{ijktw} \ \leq 4(1 - \pi_{4ktw}) \qquad \forall\, i, j, k, t, w \quad (13B)$$

Constraints (14-17)

$$\sum_{\mathrm{i=1}}^{I} \sum_{j=1}^{j_i} \eta_{ij1ktw} = 1 - \pi_{0ktw} \qquad \forall\, k, t, w \qquad (14)$$

$$\sum_{\mathrm{i=1}}^{I} \sum_{j=1}^{j_i} \eta_{ij2ktw} = 1 - \pi_{0ktw} - \pi_{1ktw} \qquad \forall\, k, t, w \qquad (15)$$

$$\sum_{\mathrm{i}=1}^{I}\ \sum_{j=1}^{j_i}\eta_{ij3ktw} = 1\ -\ \pi_{0ktw} -\ \pi_{1ktw} -\ \pi_{2ktw} \qquad \forall\, k,t,w \qquad (16)$$

$$\sum_{\mathrm{i}=1}^{I}\ \sum_{j=1}^{j_i}\eta_{ij4ktw} = 1 -\ \pi_{0ktw} -\ \pi_{1ktw} -\ \pi_{2ktw} -\ \pi_{3ktw} \qquad \forall\, k,t,w \qquad (17)$$

Constraint 18A

$$Y_{iji'j'pktw}\ \ >=\ \eta_{ijpktw} +\ \eta_{i'j'(p+1)ktw}\ -1 \qquad \forall\ \ p,k,t,w,i,j \quad (18A)$$

Constraint 19

Minimize total cost:

$$\sum_{\mathrm{w}=1}^{W}\sum_{\mathrm{i}=1}^{I}\ \sum_{\mathrm{t}=1}^{T}\left[c_i^{sub} * S_{itw} + \mathrm{c}_\mathrm{i}^{\mathrm{inv}} * I_{itw} + c_i^{pow}\ * X_{itw} + \sum_{\mathrm{j}=1}^{j_i}\sum_{\mathrm{k}=1}^{K}\ c_{ijk}^{pack} * Q_{ijktw}\right]$$

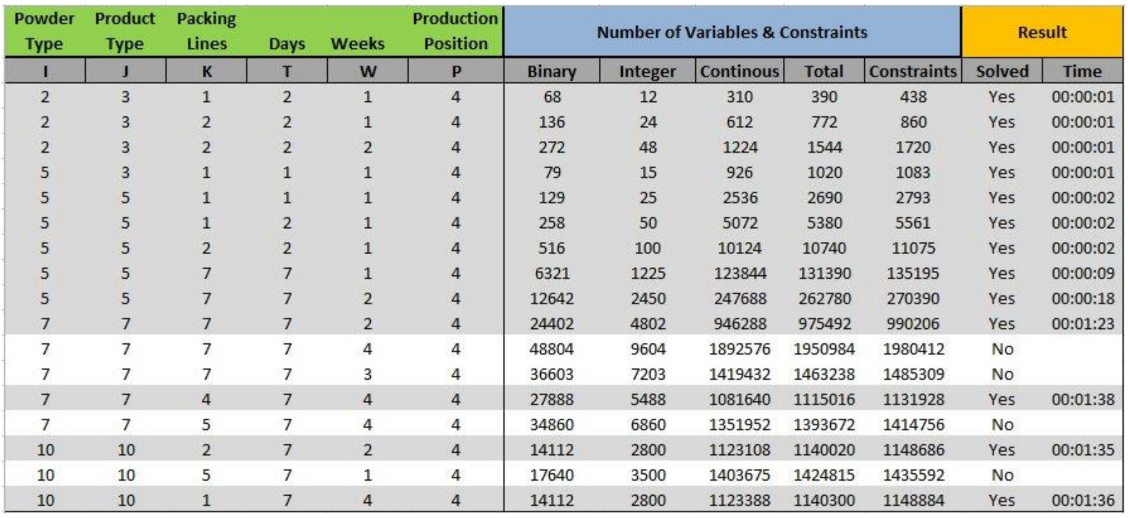
$$+\sum_{j=1}^{j_i} c_{ij}^{lost} * \left(d_{ijw} - \sum_{\mathrm{t}=1}^{T}\sum_{\mathrm{k}=1}^{K}\ Q_{ijktw}\right) \qquad (19)$$

$$+\sum_{w=1}^{W}\sum_{t=1}^{7}\sum_{k=1}^{K}\sum_{p=1}^{4}\sum_{i=1}^{I}\sum_{j=1}^{j_\mathrm{i}}\sum_{\mathrm{i'}=1}^{\mathrm{I}}\sum_{\mathrm{j'}=1}^{j_{\mathrm{i'}}} c_{\mathrm{iji'j'k}}^{setup} * Y_{iji'j'pktw}$$

3.2.3.1 Performance of MILP model

After linearization, our problem turned to a MILP problem. We solved our second model using CPLEX solver to see that its performance gets a lot better. Table 2 provides the performance of the MILP model.

Table 2. Performance of the MILP Model



Powder Type	Product Type	Packing Lines	Days	Weeks	Production Position	Number of Variables & Constraints					Result	
I	J	K	T	W	P	Binary	Integer	Continous	Total	Constraints	Solved	Time
2	3	1	2	1	4	68	12	310	390	438	Yes	00:00:01
2	3	2	2	1	4	136	24	612	772	860	Yes	00:00:01
2	3	2	2	2	4	272	48	1224	1544	1720	Yes	00:00:01
5	3	1	1	1	4	79	15	926	1020	1083	Yes	00:00:01
5	5	1	1	1	4	129	25	2536	2690	2793	Yes	00:00:02
5	5	1	2	1	4	258	50	5072	5380	5561	Yes	00:00:02
5	5	2	2	1	4	516	100	10124	10740	11075	Yes	00:00:02
5	5	7	7	1	4	6321	1225	123844	131390	135195	Yes	00:00:09
5	5	7	7	2	4	12642	2450	247688	262780	270390	Yes	00:00:18
7	7	7	7	2	4	24402	4802	946288	975492	990206	Yes	00:01:23
7	7	7	7	4	4	48804	9604	1892576	1950984	1980412	No	
7	7	7	7	3	4	36603	7203	1419432	1463238	1485309	No	
7	7	4	7	4	4	27888	5488	1081640	1115016	1131928	Yes	00:01:38
7	7	5	7	4	4	34860	6860	1351952	1393672	1414756	No	
10	10	2	7	2	4	14112	2800	1123108	1140020	1148686	Yes	00:01:35
10	10	5	7	1	4	17640	3500	1403675	1424815	1435592	No	
10	10	1	7	4	4	14112	2800	1123388	1140300	1148884	Yes	00:01:36

As it is shown in Table 2 MILP model outperforms MIQP model not only in number of variables but also in execution time. CPLEX solver is used in order to solve MILP model. CPLEX solves almost all problems at most in 2 minutes. As a very significant improvement, problems with more than 1,120,000 variables and 1,140,000 constraints can be solved with the current formulation. The largest size of the problem that can be handled with the current formulation can be expressed as 100 variables, i.e., I=J=10 with one production line and 4 weeks of planning; or else 49 products, i.e., I=J=7 with seven production lines and two weeks of planning. Noting that there are 70 products and 7 production lines in the current detergent manufacturing system, the generated MILP is capable of generating an optimal plan for 4 weeks.

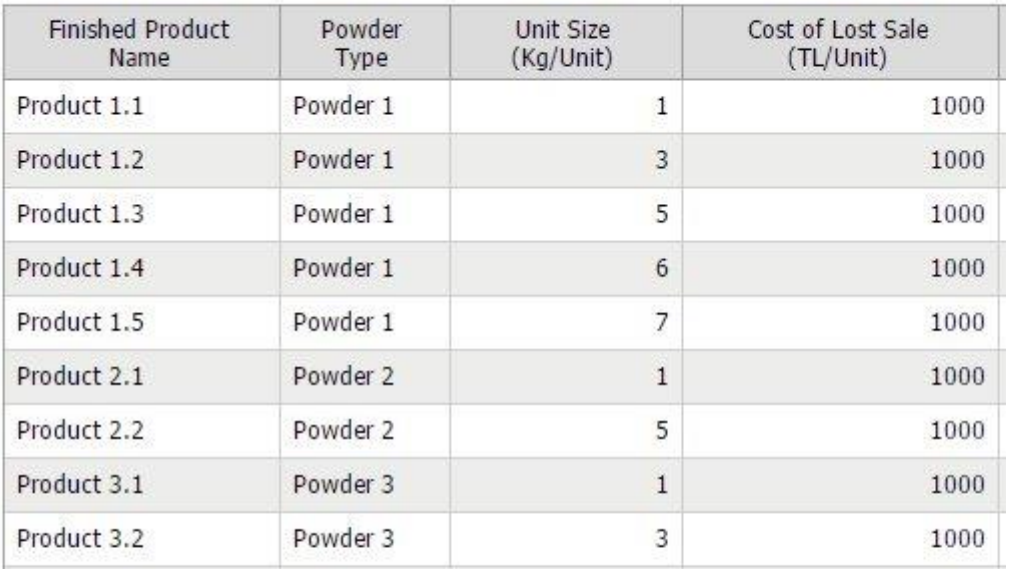
3.3 Testing scenarios

In order to verify our MILP model we developed some scenarios changing from very basic to more complicated ones and tried to gain insights. In the next section, a base scenario is created and the problem is solved based on this scenario. Then, in each section, different scenarios are developed by changing just one parameter and the results are analyzed.

3.3.1 Base scenario

In this problem setting, 9 different finished products are used produced from 3 different powders, i.e., $I = 3,\ J_1 = 5,\ J_2 = 2,\ J_3 = 2$ as seen in Table 3.

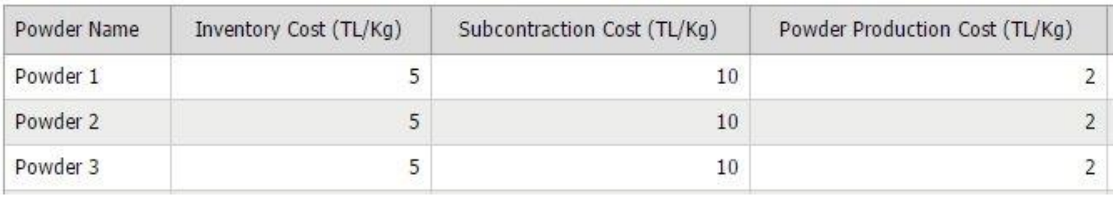
Table 3. Finished Products List



Finished Product Name	Powder Type	Unit Size (Kg/Unit)	Cost of Lost Sale (TL/Unit)
Product 1.1	Powder 1	1	1000
Product 1.2	Powder 1	3	1000
Product 1.3	Powder 1	5	1000
Product 1.4	Powder 1	6	1000
Product 1.5	Powder 1	7	1000
Product 2.1	Powder 2	1	1000
Product 2.2	Powder 2	5	1000
Product 3.1	Powder 3	1	1000
Product 3.2	Powder 3	3	1000

Finished products are produced from 3 different types of powders as seen in Table 4.

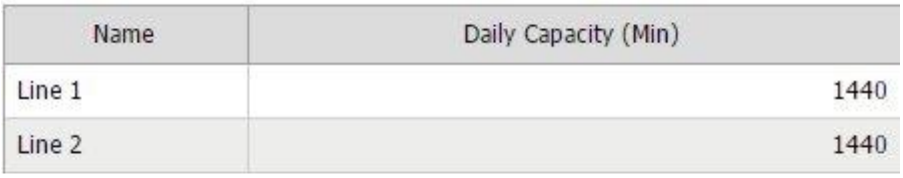
Table 4. Powder Types List



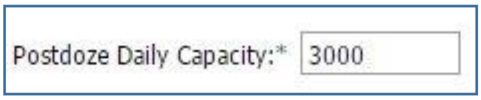
Powder Name	Inventory Cost (TL/Kg)	Subcontraction Cost (TL/Kg)	Powder Production Cost (TL/Kg)
Powder 1	5	10	2
Powder 2	5	10	2
Powder 3	5	10	2

The production lines and production capacities of base scenario can be seen in Table 5 and Figure 1.

Table 5. Production Line List



Name	Daily Capacity (Min)
Line 1	1440
Line 2	1440

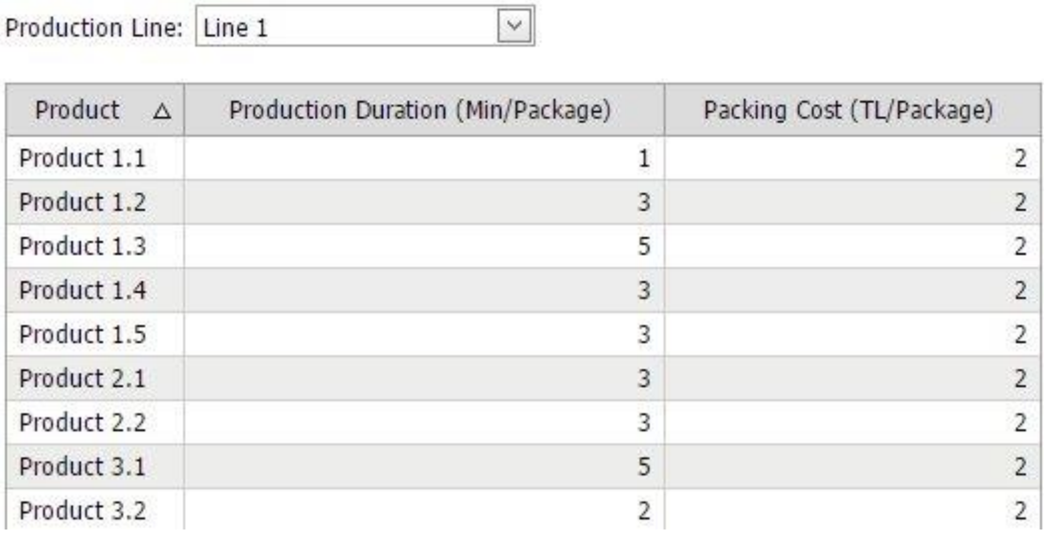


Postdoze Daily Capacity:* 3000

Figure 1. Postdoze Daily Capacity

Production Durations and Packings Costs of finished products on Line 1 and Line 2 are shown in Table 6 and Table 7 respectively.

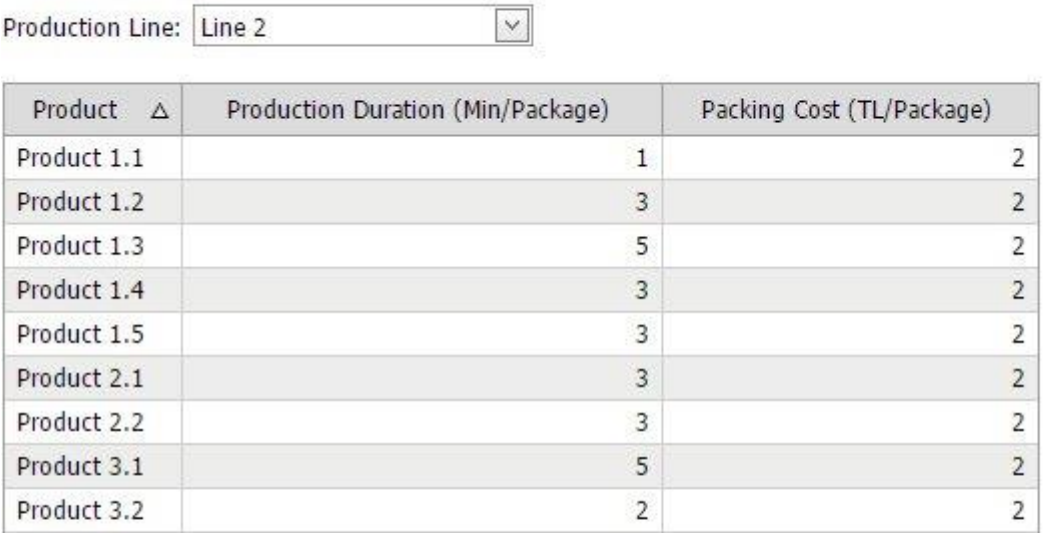
Table 6. Line 1 Production Parameters



Production Line: Line 1

Product △	Production Duration (Min/Package)	Packing Cost (TL/Package)
Product 1.1	1	2
Product 1.2	3	2
Product 1.3	5	2
Product 1.4	3	2
Product 1.5	3	2
Product 2.1	3	2
Product 2.2	3	2
Product 3.1	5	2
Product 3.2	2	2

Table 7. Line 2 Production Parameters

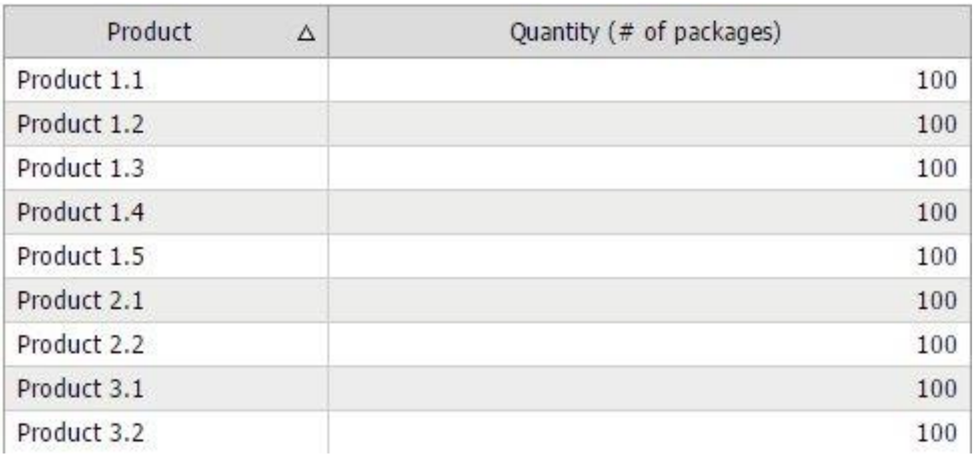


Production Line: Line 2

Product △	Production Duration (Min/Package)	Packing Cost (TL/Package)
Product 1.1	1	2
Product 1.2	3	2
Product 1.3	5	2
Product 1.4	3	2
Product 1.5	3	2
Product 2.1	3	2
Product 2.2	3	2
Product 3.1	5	2
Product 3.2	2	2

Demand Data of first week can be seen in Table 8.

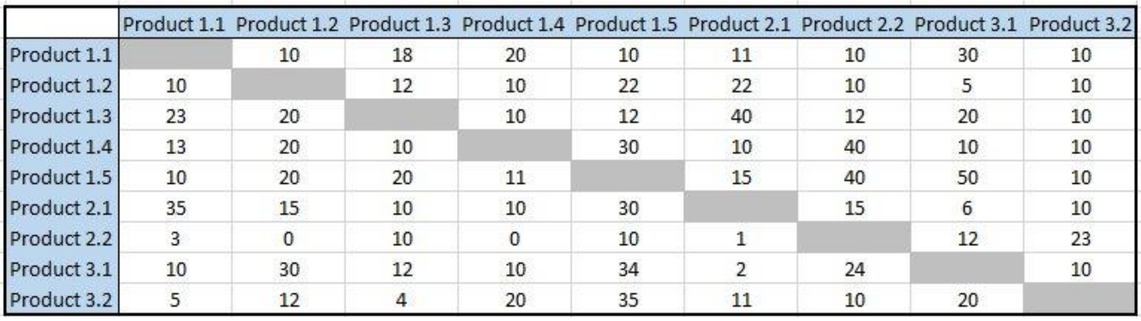
Table 8. Demand Data



Product △	Quantity (# of packages)
Product 1.1	100
Product 1.2	100
Product 1.3	100
Product 1.4	100
Product 1.5	100
Product 2.1	100
Product 2.2	100
Product 3.1	100
Product 3.2	100

Setup Costs of demanded 9 product can be seen in Table 9.

Table 9. Setup Costs



	Product 1.1	Product 1.2	Product 1.3	Product 1.4	Product 1.5	Product 2.1	Product 2.2	Product 3.1	Product 3.2
Product 1.1		10	18	20	10	11	10	30	10
Product 1.2	10		12	10	22	22	10	5	10
Product 1.3	23	20		10	12	40	12	20	10
Product 1.4	13	20	10		30	10	40	10	10
Product 1.5	10	20	20	11		15	40	50	10
Product 2.1	35	15	10	10	30		15	6	10
Product 2.2	3	0	10	0	10	1		12	23
Product 3.1	10	30	12	10	34	2	24		10
Product 3.2	5	12	4	20	35	11	10	20	

To summarize, the model is run to solve a problem that includes 9 demanded different finished products, 3 powders, 2 production lines and one day period. The resultant optimal schedule is shown in Figure 2.

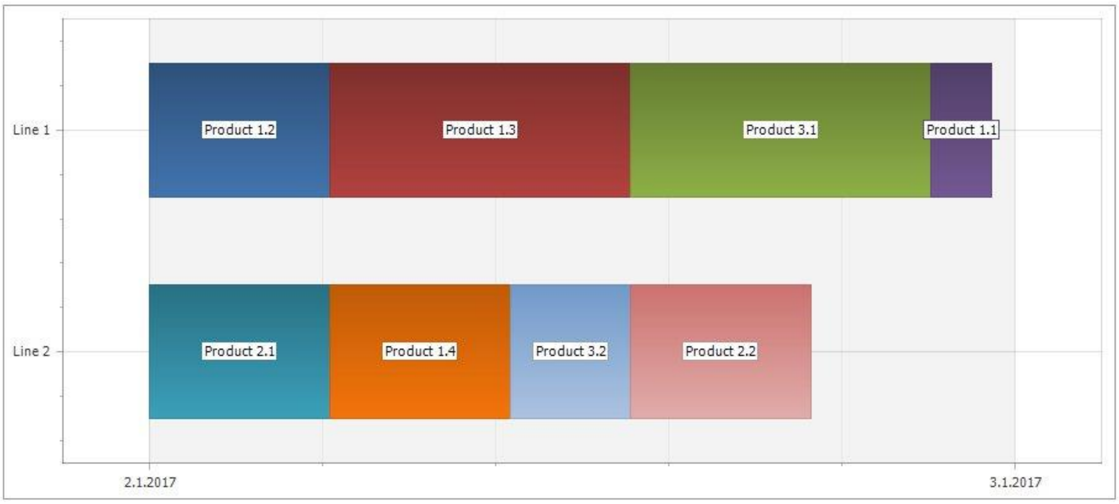
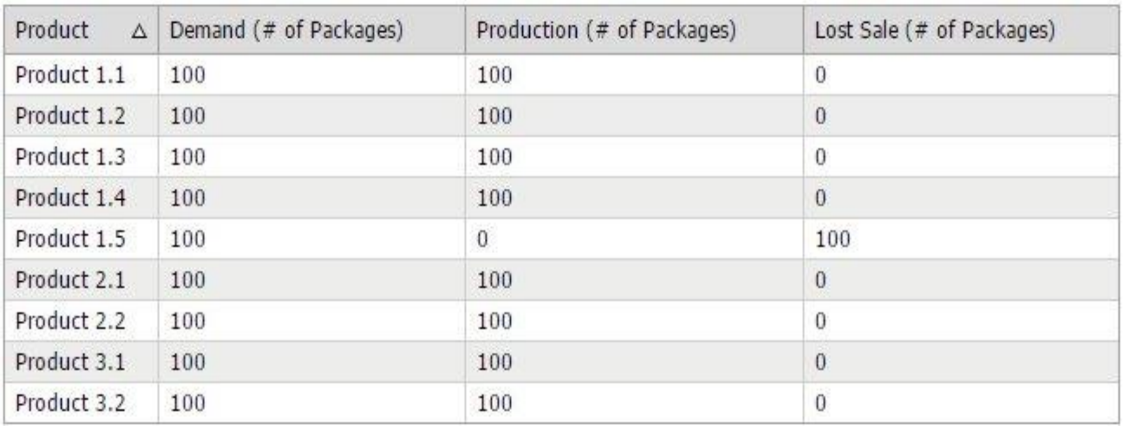


Figure 2. The schedule of base scenario

According to plan, in 2.1.2017, Line 1 should produce Product 1.2, Product 1.3, Product 3.1 and Product 1.1 respectively. Similarly, Line 2 should produce Product 2.1, Product 1.4, Product 3.2 and Product 2.2. The width of each color shows the allocated time of each productions. This allocated time is calculated by multiplying the unit production duration of each finished product and the quantity to be produced.

The model tries to satisfy the demand of each product in order to avoid high lost sale costs. However, as stated before, the production lines are allowed to produce at most 4 different products in a day. Therefore, total 8 finished products are produced and the demand of 1 product is not satisfied. Table 10 shows the summary of this period in terms of demanded, produced and lost quantities.

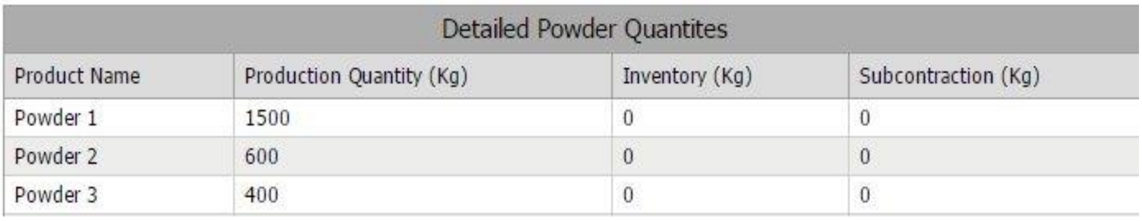
Table 10. The Summary Report of First Week



Product △	Demand (# of Packages)	Production (# of Packages)	Lost Sale (# of Packages)
Product 1.1	100	100	0
Product 1.2	100	100	0
Product 1.3	100	100	0
Product 1.4	100	100	0
Product 1.5	100	0	100
Product 2.1	100	100	0
Product 2.2	100	100	0
Product 3.1	100	100	0
Product 3.2	100	100	0

The model also do production planning to optimize the powder quantities that will be used in the production of finished products. The results of production planning is shown in Table 11.

Table 11. Production Planning of First Week



Detailed Powder Quantites			
Product Name	Production Quantity (Kg)	Inventory (Kg)	Subcontraction (Kg)
Powder 1	1500	0	0
Powder 2	600	0	0
Powder 3	400	0	0

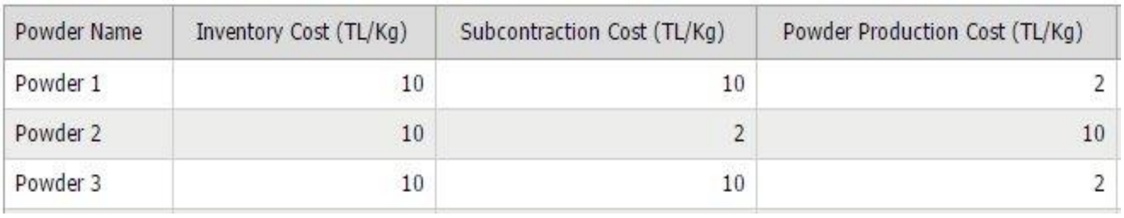
The columns of Production Quantity, Inventory and Subcontracting shows how much powder is produced, inventoried and subcontracted respectively. Since the plan is made for one week period, holding inventory is not preferred in any powder type. Subcontracting is also not preferred because it is more costly then production.

In the next sections, alternative scenarios are handled.

3.3.2 Scenario 1: Decreasing the unit subcontracting cost

In the first scenario, subcontracting is not preferred due to high unit subcontracting costs. But, what would happen if subcontracting is less costly then production? In order to analyze, subcontracting and production costs of Powder 2 are interchanged. The new values can be seen in Table 12.

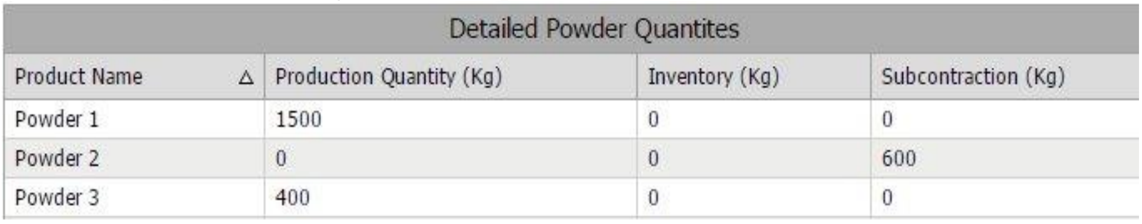
Table 12. Optimal Subcontracting and Production Costs in Scenario 1



Powder Name	Inventory Cost (TL/Kg)	Subcontraction Cost (TL/Kg)	Powder Production Cost (TL/Kg)
Powder 1	10	10	2
Powder 2	10	2	10
Powder 3	10	10	2

After this change, the model is run again and products are rescheduled. New weekly powder report can be seen in Table 13.

Table 13. Optimal Weekly Powder Report in Scenario 1

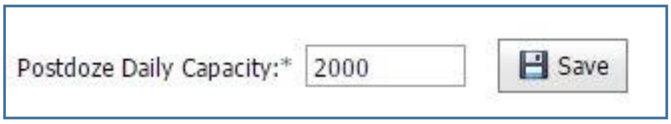


Detailed Powder Quantites			
Product Name △	Production Quantity (Kg)	Inventory (Kg)	Subcontraction (Kg)
Powder 1	1500	0	0
Powder 2	0	0	600
Powder 3	400	0	0

Table 13 shows that powder 2 is obtained by subcontracting instead of production which is very intuitive.

3.3.3 Scenario 2: decreasing the postdoze capacity

Firstly, production and subcontracting costs are changed to original values as in Table 2. Then the daily capacity of Postdoze which produces the powders is decreased to 2000 kg/day as in Figure 3.

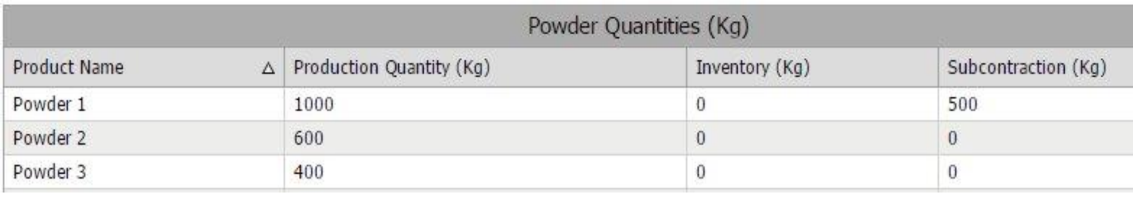


Postdoze Daily Capacity:* 2000

Figure 3. Decreased postdoze capacity in scenario 2

The new Powder Report can be seen in Table 14.

Table 14. Optimal Powder Report in Scenario 2



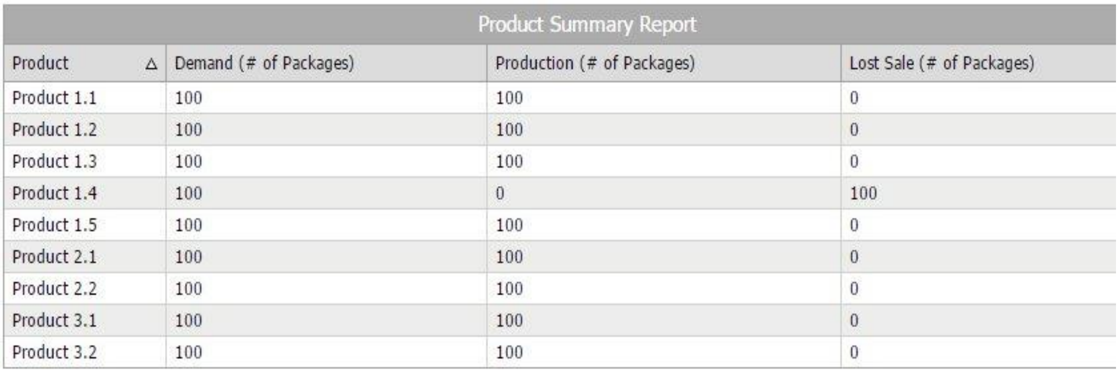
Powder Quantities (Kg)			
Product Name △	Production Quantity (Kg)	Inventory (Kg)	Subcontraction (Kg)
Powder 1	1000	0	500
Powder 2	600	0	0
Powder 3	400	0	0

Obviously, the sum of the powder production quantities in Table 14 is decreased to 2000. Since daily postdoze capacity (2000 kg) does not meet the demanded quantity, subcontraction is preferred in powder 1.

3.3.4 Scenario 3: Increasing the unit lost sale cost

Our aim in this scenario is to test the effect of unit lost sale cost. In the base scenario, unit lost sales costs are equal for all products as shown in Table 2. In the optimal solution of the base scenario, demand of Product 1.5 is not satisfied. In order test the effect of unit lost sale cost, the unit lost sale cost of Product 1.5 is increased to 1000 from 5000, and the products are rescheduled. The new Product Summary Report is shown Table 15.

Table 15. Product Summary Report with Updated Lost Sale Cost in Scenario 3



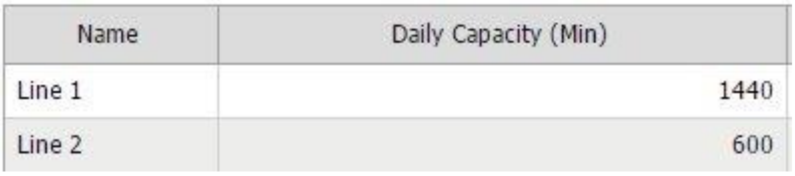
Product Summary Report			
Product △	Demand (# of Packages)	Production (# of Packages)	Lost Sale (# of Packages)
Product 1.1	100	100	0
Product 1.2	100	100	0
Product 1.3	100	100	0
Product 1.4	100	0	100
Product 1.5	100	100	0
Product 2.1	100	100	0
Product 2.2	100	100	0
Product 3.1	100	100	0
Product 3.2	100	100	0

By considering Table 10, it can easily be realized that the model chooses to produce Product 1.5 that has higher unit lost sale cost.

3.3.5 Scenario 4: decreasing the capacity of a production line

In order to test the effect of decreasing the production capacity, we decreased the daily capacity of Line 2 to a smaller value as is Table 16.

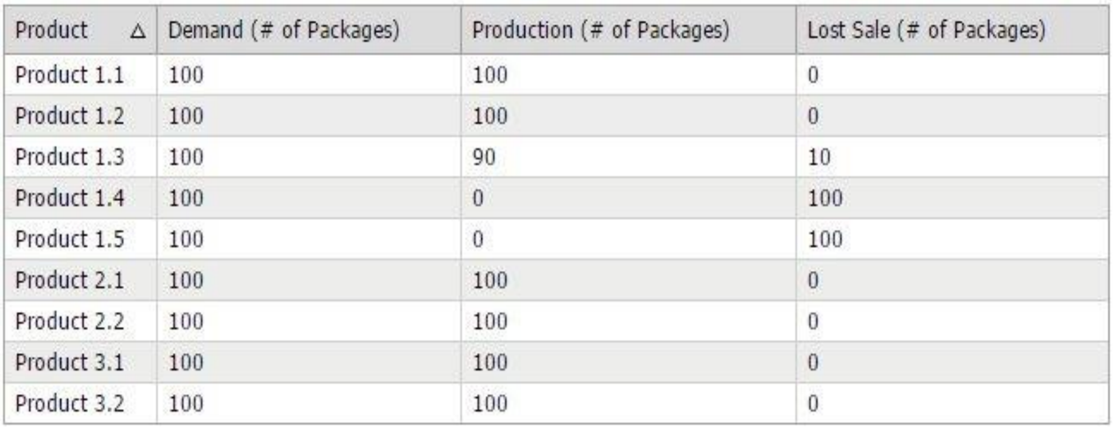
Table 16. Line Capacities in Scenario 4



Name	Daily Capacity (Min)
Line 1	1440
Line 2	600

Then we reschedule the products. The optimal Product Summary Report can be seen in Table 17.

Table 17. Product Summary Report with Low Line Capacity in Scenario 4



Product △	Demand (# of Packages)	Production (# of Packages)	Lost Sale (# of Packages)
Product 1.1	100	100	0
Product 1.2	100	100	0
Product 1.3	100	90	10
Product 1.4	100	0	100
Product 1.5	100	0	100
Product 2.1	100	100	0
Product 2.2	100	100	0
Product 3.1	100	100	0
Product 3.2	100	100	0

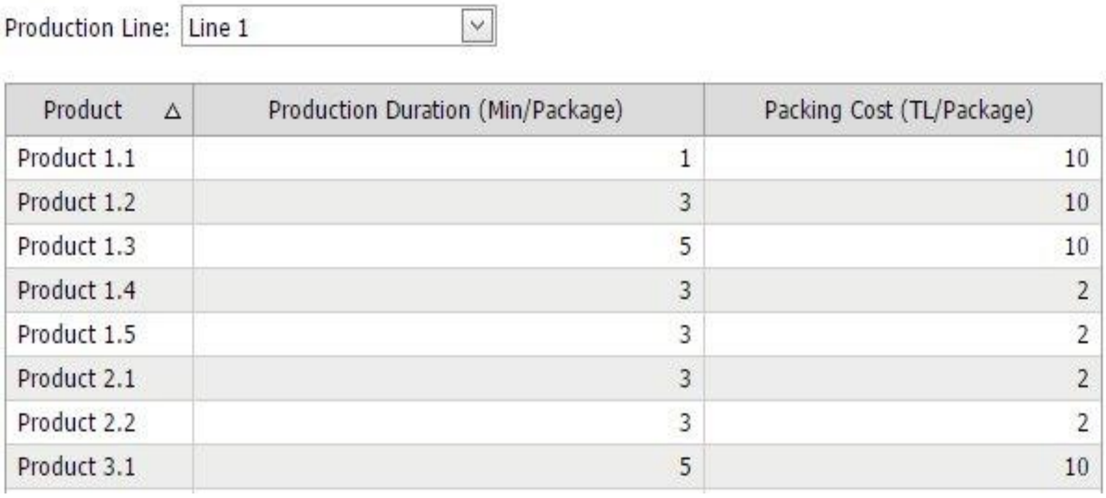
When compared to Base Scenario results in Table 7, it can be seen that the demand of Product 1.3 and Product 1.4 are not satisfied in addition to Product 1.5. Since capacity is not enough, the model has to stop producing some products.

Product 1.3 and Product 1.4 are chosen according to their production durations. (Table 4, Table 5). Since the goal of the model is cost minimization and lost cost is calculated by the number of unsatisfied number of packages, the model prefers not to produce the products that have longer production durations in order to enhance the number of packages to be produced. At this point, it should be noted that this explanation is valid for equal lost sale costs.

3.3.6 Scenario 5: Increasing the unit packing cost

In this scenario the goal is to test the reaction of our model to increasing the unit packing cost of finished products on a line. For testing purposes, the packing costs of products that are produced on line 1 according to base scenario, are increased. The new values can be seen in Table 18.

Table 18. Production Variables with Updated Packing Costs in Scenario 5



Production Line: Line 1

Product △	Production Duration (Min/Package)	Packing Cost (TL/Package)
Product 1.1	1	10
Product 1.2	3	10
Product 1.3	5	10
Product 1.4	3	2
Product 1.5	3	2
Product 2.1	3	2
Product 2.2	3	2
Product 3.1	5	10

When these values are compared with the base scenario parameters in table 7 we see that unit packing costs of the Product 1.1, Product 1.2, Product 1.3 and Product 3.1 in Line 1 are increased.

The model is run again and the new schedule can be seen in Figure 4.

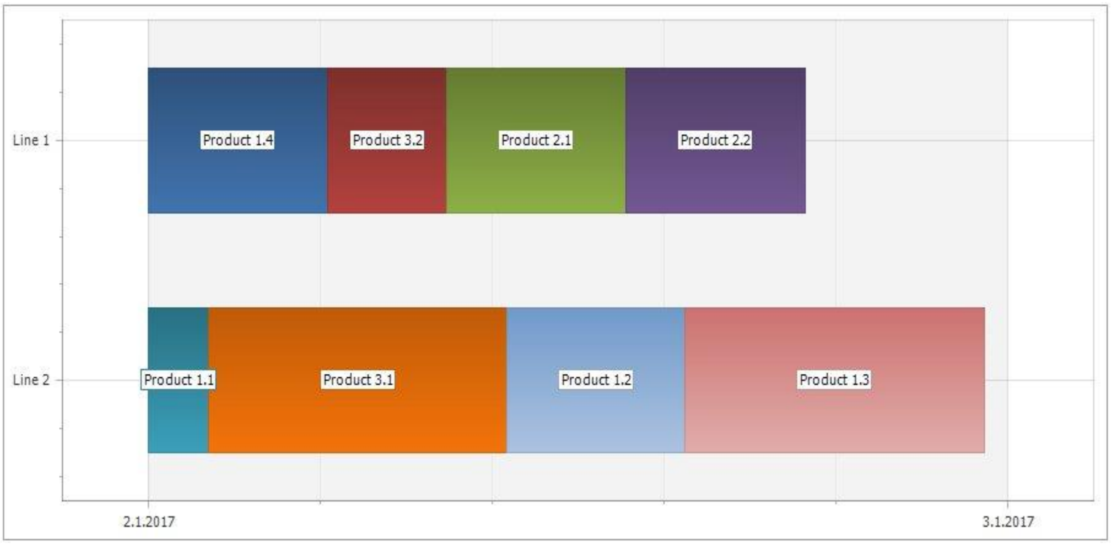


Figure 4. Optimal schedule according to updated packing values in scenario 5

As expected, in order to reduce the packing cost, Product 1.1, Product 3.1, Product 1.2 and Product 1.3 are assigned to Line 2 rather than Line 1.

3.3.7 Scenario 6: Increasing the Setup Costs

In this scenario, the aim is to analyze the effect of setup costs between productions of two consecutive products. In order to test, the setup costs for switching production from Product 3.1 and Product 1.1, and from Product 1.2 and Product 1.3 are doubled as 20 and 20. The new schedule can be seen in Figure 3.

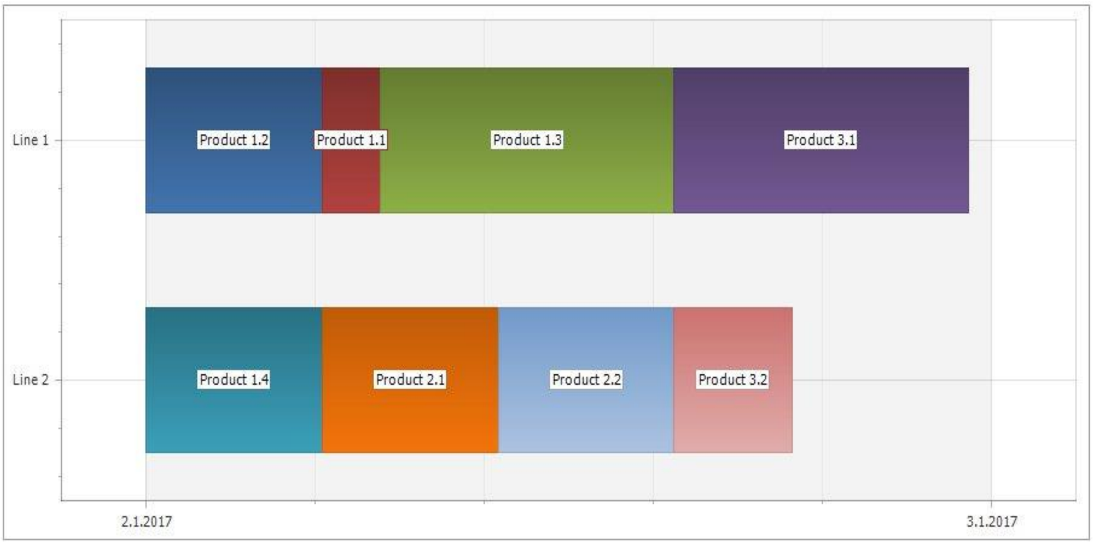


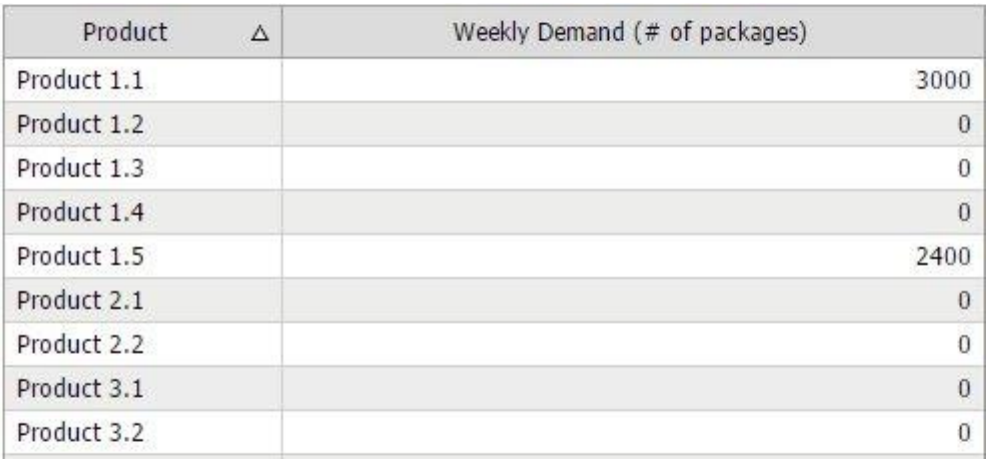
Figure 5. Schedule according to updated setup costs in scenario 6

We see in figure 5 that the optimal production sequence is changed according to the changing values of setup costs. Note that in the base schedule in Figure 1, Line 1 production sequence is Product (1,2), (1,3), (3,1), (1,1) and Line 2 production sequence is Product (2,1), (1,4), (3,2), (2,2). Since the sequence dependent setup cost from product (3,1) to product (1,1) is very costly in Scenario 6, Product (1,1) is not assigned after product (3,1). Similarly, Product 1.2 and Product 1.3 is not produced consecutively in order to minimize cost.

3.3.8 Scenario 7: Daily inventory holding

In this final scenario, the aim is to force the model to hold daily inventory. In the setting of this scenario, the only active packing line is Line 1 and the demand data is updated as in Table 19.

Table 19. Updated Demand Data in Scenario 7

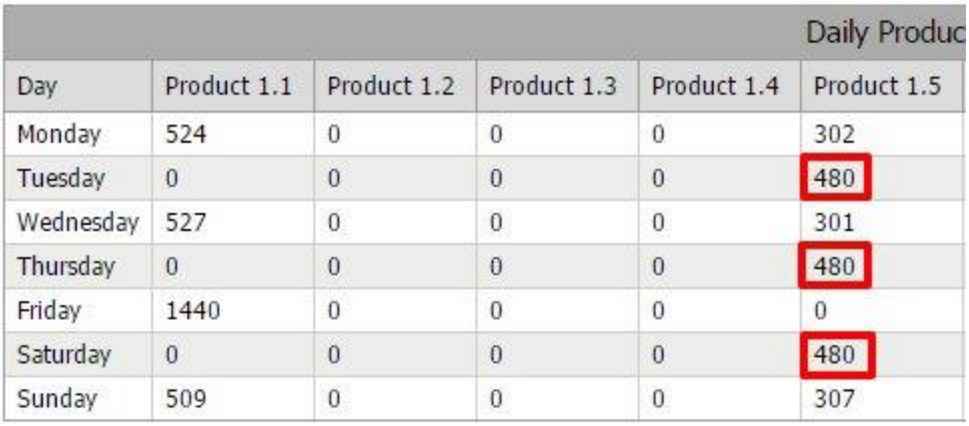


Product △	Weekly Demand (# of packages)
Product 1.1	3000
Product 1.2	0
Product 1.3	0
Product 1.4	0
Product 1.5	2400
Product 2.1	0
Product 2.2	0
Product 3.1	0
Product 3.2	0

When the values in Table 19 is checked, it will be realized that producing 3000 packages of Product 1.1 will take about two days. Therefore, Product 1.5 is needed to be produced average 480 packages during five days. This condition is possible is according to Line Capacity (number of producing 1 package takes 3 min. as in Table 6). However, (since the package size of Product 1.5 is 7), the needed quantity of powder 480*7=3360 is above the daily postdoze capacity. In this regard, it is expected that the needed quantity of Powder 1 which cannot be produced in the same day, will be produced and in another day.

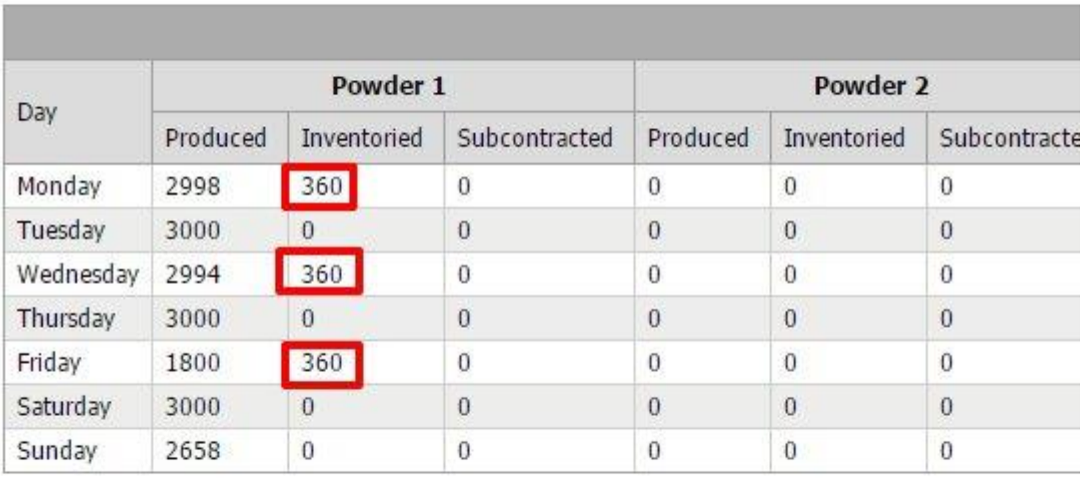
The daily product report can be seen in Table 20 and daily powder report can be seen in Figure 21.

Table 20. Daily Product Report in Scenario 7



Daily Produc					
Day	Product 1.1	Product 1.2	Product 1.3	Product 1.4	Product 1.5
Monday	524	0	0	0	302
Tuesday	0	0	0	0	480
Wednesday	527	0	0	0	301
Thursday	0	0	0	0	480
Friday	1440	0	0	0	0
Saturday	0	0	0	0	480
Sunday	509	0	0	0	307

Table 21. Daily Powder Report in Scenario 7



Day	Powder 1			Powder 2		
	Produced	Inventoried	Subcontracted	Produced	Inventoried	Subcontracte
Monday	2998	360	0	0	0	0
Tuesday	3000	0	0	0	0	0
Wednesday	2994	360	0	0	0	0
Thursday	3000	0	0	0	0	0
Friday	1800	360	0	0	0	0
Saturday	3000	0	0	0	0	0
Sunday	2658	0	0	0	0	0

When we examine two figures, it can be realized that before the days which the quantity of Product 1.5 is 480 (Table 20), Powder 1 is inventoried (Table 21). It can be concluded that the model holds inventory when it is needed. It should also be noted that Subcontracting is not preferred as in Scenario 2. Because it is more costly than inventory holding. (Table 4)

CHAPTER 4

DEVELOPMENT OF DECISION SUPPORT SYSTEM FOR THE INTEGRATED MODEL

Turban and Aronson (2001) define Decision Support System (DSS) as a model-based or knowledge-based system intended to support managerial decision making in semi structured or unstructured situations. In this study, A DSS is developed in order to make our model easy to apply into practical problems. Thanks to this DSS, a planner can manage all the production planning and scheduling processes without any mathematical or modeling background.

According to Janakiraman and Sarukesi (2004), a DSS in comprised of three subsystems that work together to create a complete DSS. Those three components are the Data Management Subsystem, Model Management Subsystem and Dialogue Management System. The database management system restore, organize and retrieve data to be used in the decision making process. Model management system contains the model which use mathematical techniques to describe real problem. And finally, the dialog management system enable user to manipulate data and to reach management system. The dialogue management subsystem takes the form of a graphical user interface, presenting the relevant information and courses of action to the user by making calls to the model management and data management subsystems. (Janakiraman and Sarukesi, 2004, p. 29).

In this study, a web-based DSS is developed. In web-based systems, the application is hosted on a server and clients use the application by the help of a thin client web browser like Internet Explorer, Google Chrome, etc. In today's world, it is an undeniable fact that web-based applications are very practical. Still, it is important to emphasize the advantages of web-based decision support systems.

Firstly, Web-based DSS makes easier and less costly to make decision relevant information accessible to all planners and managers in geographically distributed locations (Power, 1998). By the use of World Wide Web (WWW) infrastructure, users from whole over the world can reach and even edit data from different locations.

Secondly, Web provides remote control for managers to observe the decision making processes. Therefore, web-based DSSs are more widely used and controlled. An enormous amount of data can be gathered routinely with the help of web-bases DSSs. Thus, companies can manage knowledge repository easily.

Lastly, the web based systems are also more preferable because they provide ability to communicate with other applications like ERP systems. This feature enables companies to exchange information between systems.

In this section, the Decision Support System which helps planner for production planning and scheduling will be explained in terms of framework, database and development.

4.1 DSS framework and information flow

Database component of our DSS is a relational database including the relevant master data (product information, production line information etc.), inputs (demands, model parameters on unit costs, durations, capacity, etc.) and outputs of the model (optimal production plan, schedule, quantities, etc.). SQL Server 2014 is used as a Relational Database Management System, RDMS.

The model base includes a MILP model that optimizes production plan and schedule. This model (in APPENDIX) runs on GAMS software with the help of GUI.

Finally, the GUI is developed on the .NET framework 4.0 environment using C# language. This web-based system enables users to insert, edit and delete data to the database very easily. GUI also helps users to run the model on GAMS software and present the results in charts and tables. The component structure of DSS is shown in Figure 6.

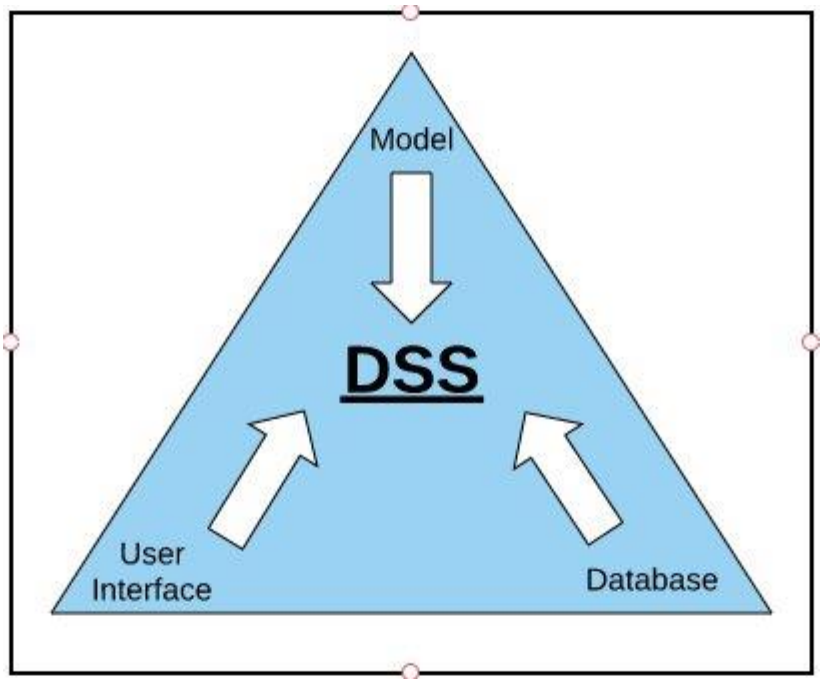


Figure 6. Schematic view of DSS

The use of DSS includes three basic steps. Firstly, the master data should be defined to system. The term master data contains the information which is needed for model to work, namely, powder types, finished product types, production lines. Secondly, the variables values (inputs) like setup costs and demands should be defined. Finally, user should run the model by setting the time span for schedule. Model runs on GAMS Solver and results are exported to CSV files. If the exit code that is sent from GAMS to GUI shows the success of solver, then, by the help of GUI, results are saved to the database. After this step, user can see the outputs of GAMS using charts or tables. The use case diagram in Figure 7 shows the data flow.

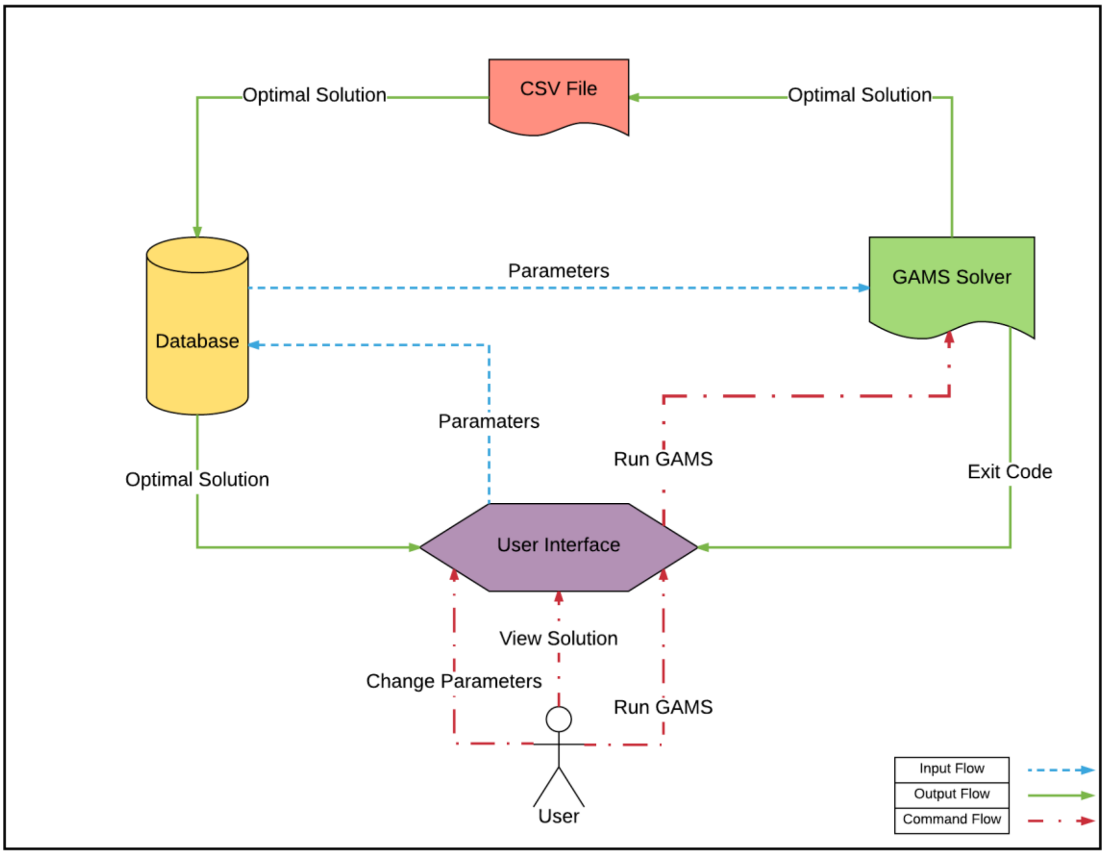
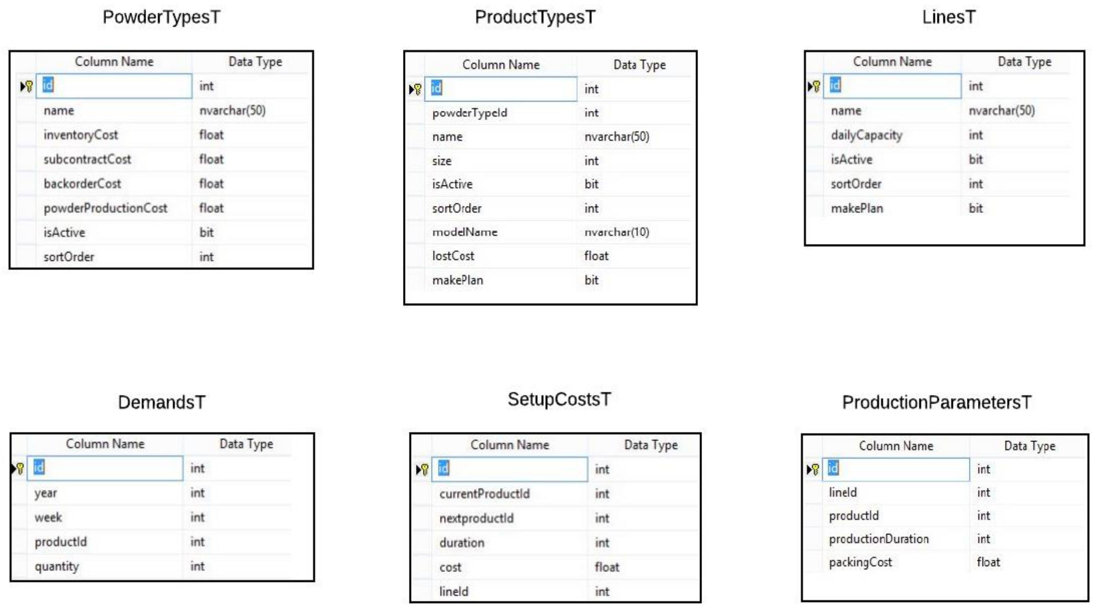


Figure 7. Information flow in DSS

4.2 DSS database

Database is one of the core element of a DSS because as French et al. stated (2009) different types of data, information and knowledge are needed and dominate the decision making processes in different types of decision situations. It is clear that understanding and managing the different types of data, information and knowledge has a tremendous effect on effectiveness of DSS. In this regard, Relational Database Management System (RDMS) and more specifically Microsoft Sql Server (MSSQL) Server is preferred for database management. RDMSs are appropriate for DSSs because they enable designer to generate relations schemes and prevent unnecessary redundant data. Additionally, storing and manipulating data is easy and effective in terms of both usage and performance.

In database design, meaningful table and field names are tried to be used. The tables in our DSS can be divided into 3 categories, namely, input tables, output tables and temporary tables. Input tables store master data related with powders, products, production lines and other parameter values like demand, unit costs and setup data. The designs of Input Tables can be seen in Figure 8.



PowderTypesT

Column Name	Data Type
id	int
name	nvarchar(50)
inventoryCost	float
subcontractCost	float
backorderCost	float
powderProductionCost	float
isActive	bit
sortOrder	int

ProductTypesT

Column Name	Data Type
id	int
powderTypeId	int
name	nvarchar(50)
size	int
isActive	bit
sortOrder	int
modelName	nvarchar(10)
lostCost	float
makePlan	bit

LinesT

Column Name	Data Type
id	int
name	nvarchar(50)
dailyCapacity	int
isActive	bit
sortOrder	int
makePlan	bit

DemandsT

Column Name	Data Type
id	int
year	int
week	int
productId	int
quantity	int

SetupCostsT

Column Name	Data Type
id	int
currentProductId	int
nextproductId	int
duration	int
cost	float
lineId	int

ProductionParametersT

Column Name	Data Type
id	int
lineId	int
productId	int
productionDuration	int
packingCost	float

Figure 8. Input tables

The field named ‘isActive’ in the PowderTypesT, ProductTypesT and LinesT is added in order to manage delete operations. ‘isActive’ field is a Boolean field that shows the record is actively used or not. The aim of adding this of field is not to lose information even user deletes the powder, product or line type. After delete operation the field ‘isActive’ gets false value and thus, user does not see, but the record is kept in background.

The sortOrder field is added because the presentation of records is different in model and relational database. To put it more clearly, *i* and *j* are the indices that symbolize the products together. However, in relational database systems, generally one primary key field is preferred for record uniqueness. The sortOrder field helps to DSS for converting data presentation to a model compatible format.

The fields named makePlan in the ProductTypesT and LinesT is added in order provide flexibility to user for including or not excluding some products or production lines from schedule process. To illustrate, because of a machine breakdown user may prefer not to use the production line in schedules for a short time period. In this type of conditions, user should edit the field ‘makePlan’. This field’s caption is ‘Include in Plan’ in master data pages.

The data in the input fields are transferred to GAMS Solver by the help of SQL2GMS tool. This tool is an extension of GAMS Solver that reads data from an SQL Server. When a username and password is provided in the GAMS file, this tool can connect to SQL Server and read data. Thus, data transformation from SQL input tables to GAMS is easily occurs.

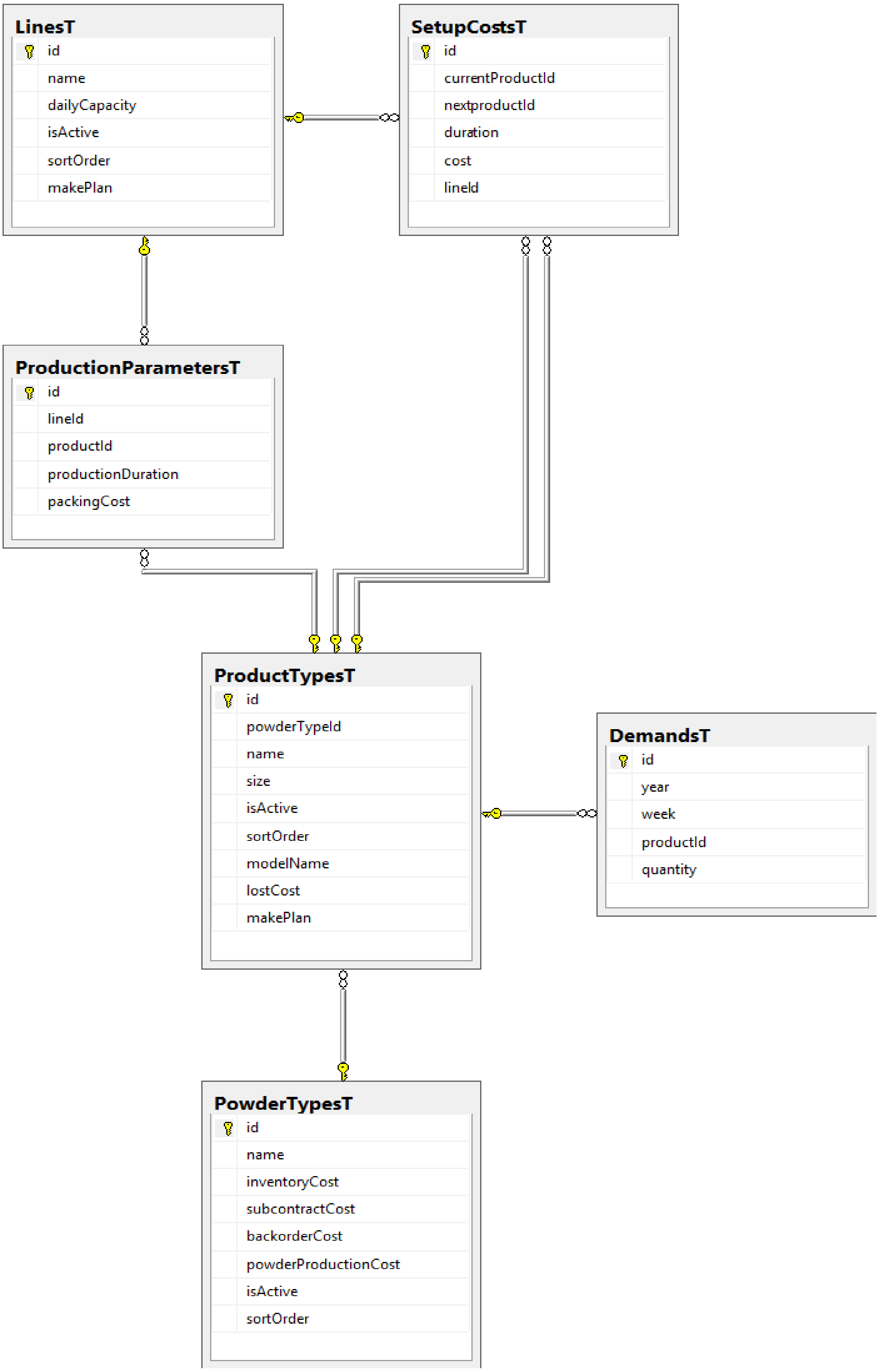
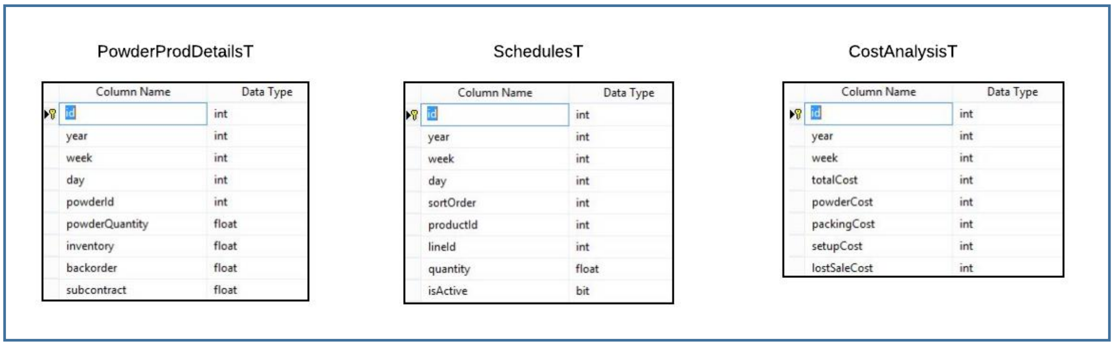


Figure 9. ER diagram for input tables

Output tables are created to contain outputs retrieved from GAMS. One of the most important output table in this DSS is SchedulesT table. This table stores production date based on product type, year, week, day, line and sort order. Output Tables can be seen in Figure 10.



PowderProdDetailsT

Column Name	Data Type
id	int
year	int
week	int
day	int
powderId	int
powderQuantity	float
inventory	float
backorder	float
subcontract	float

SchedulesT

Column Name	Data Type
id	int
year	int
week	int
day	int
sortOrder	int
productId	int
lineId	int
quantity	float
isActive	bit

CostAnalysisT

Column Name	Data Type
id	int
year	int
week	int
totalCost	int
powderCost	int
packingCost	int
setupCost	int
lostSaleCost	int

Figure 10. Output tables

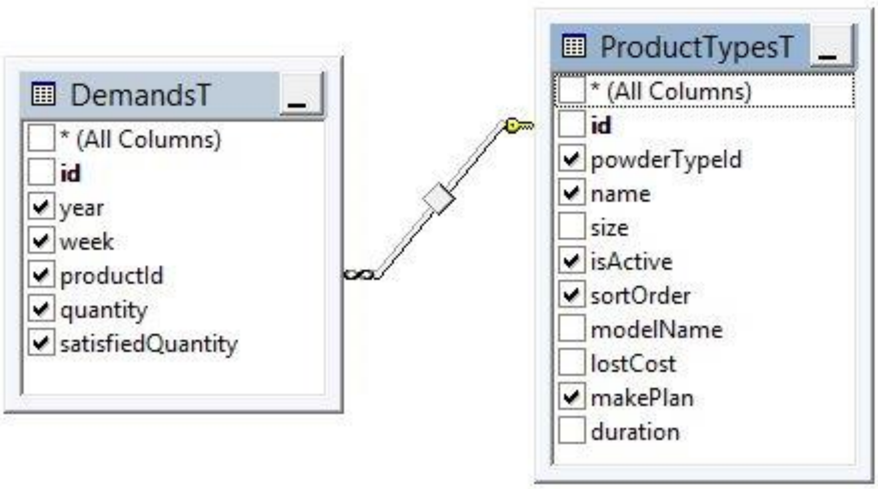
When GAMS solves the problem, it exports data into CSV files using GDXViewer tool. This step is very easy to apply using GAMS Solver. However, the comma-separated values in the CSV files are needed to export to SQL Databases in appropriate format. To handle this problem, a C# function that reads data from CSV file and converts in to SQL is added. In this process, temporary tables are needed because data structures are different. As stated before, product types are needed to be converted to an appropriate SQL table structure. Also, GAMS does not have a date type variable. However, our DSS presents results according to date. Date values are obtained by combining week and day variables. In addition to data format problems, for robustness of data, few different outputs of GAMS are stored just one table in SQL. In order to succeed this operation, Stored Procedures (SP) are applied. SPs can be defined as a group of SQL queries that can be reused by calling its special name.

To illustrate with our case, produced, subcontracted and inventoried quantities of powders are exported to the temporary tables named PowderQuantitiesT, PowderSubcontractsT and PowderInventoriesT tables. Then, an

SP called AddPowderInfo reads data from those tables and updates the powder quantity, inventory and subcontract columns of PowderProdDetailsT.

DSS asks for the week number before running the GAMS, the system save this week data into a setting table. Then, when importing results from GAMS output files, variables are saved according to their dates.

In addition to tables and SPs, Views are also widely used in DSS because views are very practical for presentational purposes. A View can be defined as a result set of a stored query. Views give opportunity to combine data from different tables in an effective way. For example, in table named DemandsT the demand data of all products are stored. However, as we stated before, user may want not to include a finished product to plans. In this type of conditions, DemandsV view enable user to query the product information and demand data with one query easily. The structure of DemandsV can be seen in Figure 14. The relation between DemandsT table ProductTypes are provided through the id column of ProductTypesT and productId columns of DemandsT since both column store the same data.



DemandsT

□ * (All Columns)
□ **id**
☑ year
☑ week
☑ productId
☑ quantity
☑ satisfiedQuantity

ProductTypesT

□ * (All Columns)
□ **id**
☑ powderTypeId
☑ name
□ size
☑ isActive
☑ sortOrder
□ modelName
□ lostCost
☑ makePlan
□ duration

Figure 11. Demandsv structure

4.3 DSS development

According to Şeref et al. (2007) a DSS developing process includes six steps, namely, Application Overview, Worksheets, User Interface, Procedures, Resolve Options and Testing and Final Packaging.

The first step, application overview is the most critical step according to the Şeref et al. (2007). In this step, the whole process and information flow is handled. Some questions should be replied in this part.

- What are inputs the DSS needs?
- How does the process start?
- How the solution will be reached?
- What will be the outputs?

In our case, the inputs are data about products, production lines, production, subcontracting and inventory holding costs, setup costs between each product change and demand data. After the inputs are provided, user can run the model with the help graphical user interface and results is obtained in a few minutes according to the problem size. The main output of our system is a schedule for finished products and cost report. Also, our DSS system store historical data about the previous weeks' productions and provide reports related with finished and semi-finished productions.

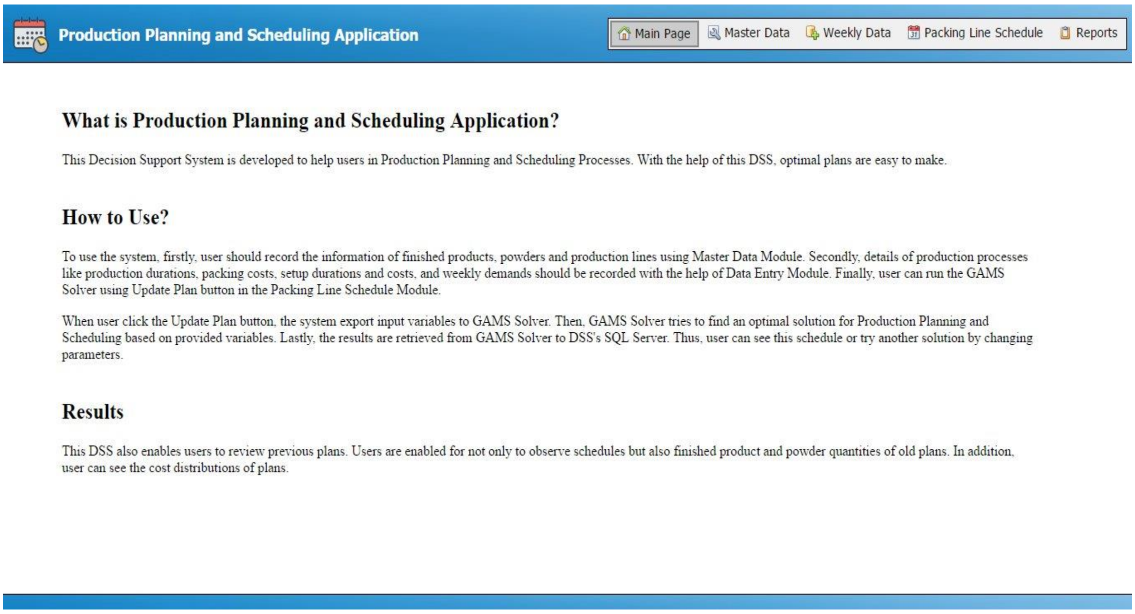
For the worksheets step, the developer should consider how many worksheets will be used and what features the worksheets should have. Worksheets part is also critical because worksheets help user to enter data to the system. Sheets are also used for output presentation. Therefore developers also decide on how they will present the results. Charts and special types of worksheets should be developed at this stage.

The third step of DSS development is Graphical User Interface (GUI). According to Galitz (1985), the aim of GUI design is developing screen layouts and interfaces that are easy to use and visually attractive. The GUI enables user to manipulate data, run model and observe the outputs without any technical skills. It is very important to have a simple, effective and user friendly GUI for not only simplify user’s job but also enhance the use of DSS. In this regard, complete understanding of users’ needs and capabilities play a critical role. Menus, icons, buttons should allow the user communicate with the program visually. Also, according to Power (2002), a well-designed GUI can increase human processing speed, reduce errors, increase productivity and create a sense of user control.

The GUI of our subject DSS is developed by considering all those critical points. The GUI starts with a Home Page.

4.3.1 Home page and DSS navigation

When we open the DSS, Main Page is displayed first as in Figure 12.



What is Production Planning and Scheduling Application?

This Decision Support System is developed to help users in Production Planning and Scheduling Processes. With the help of this DSS, optimal plans are easy to make.

How to Use?

To use the system, firstly, user should record the information of finished products, powders and production lines using Master Data Module. Secondly, details of production processes like production durations, packing costs, setup durations and costs, and weekly demands should be recorded with the help of Data Entry Module. Finally, user can run the GAMS Solver using Update Plan button in the Packing Line Schedule Module.

When user click the Update Plan button, the system export input variables to GAMS Solver. Then, GAMS Solver tries to find an optimal solution for Production Planning and Scheduling based on provided variables. Lastly, the results are retrieved from GAMS Solver to DSS's SQL Server. Thus, user can see this schedule or try another solution by changing parameters.

Results

This DSS also enables users to review previous plans. Users are enabled for not only to observe schedules but also finished product and powder quantities of old plans. In addition, user can see the cost distributions of plans.

Figure 12. Main page

At the main page center, there is a short introduction of the DSS. Also a basic explanation of the process exists. We can see the menu at the top right side. Menu has 4 main elements apart from Main Page: Master Data, Weekly Data, Packing Line

Schedule, and Reports. We can call them as modules. Each module has its own submenu that becomes visible after the first click of any menu item at the top left of the page. This submenu is added in order to ease navigation within a module. Menu Bar always stays on the screen and enables user to navigate between different modules. Users can move to different pages by just clicking on the appropriate texts.

4.3.2 Master data module

Master Data Module is developed in order to enable user to manipulate the basic data about the system. Master Data Module consists of 5 components as can be seen from the Master Data Submenu. (Figure 13)

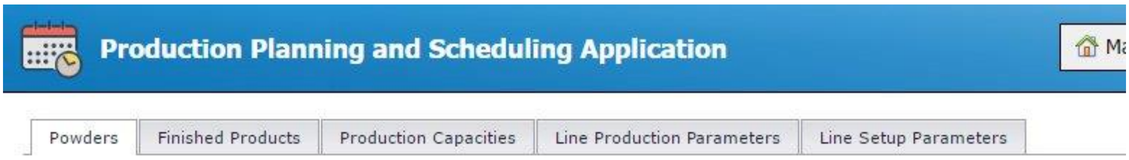

Production Planning and Scheduling Application
Ma
Powders
Finished Products
Production Capacities
Line Production Parameters
Line Setup Parameters

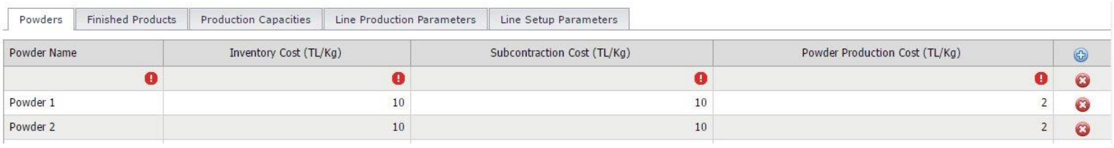
Figure 13. Master data submenu

The component names are: Powders, Finished Products, Production Capacities, Line Production Parameters, Line Setup Parameters. Those pages are added in order to manipulate master data that changes rarely. Before providing the details of the pages, it should be noted that the ASPxGridView Control of Developer Express Company is used to provide main functions in those pages. ASPxGridView is a graphical control element that shows a tabular view of data similar to excel sheets. ASPxGridView enables user to manipulate data. The use of ASPxGridView is always the same within all pages. Therefore, one general expression is provided.

Use of ASPxGridView

The columns of ASPxGridView store data related with the information stated in the header. The last column is called Command Column that allows to manage changes.

The ⊕ icon symbolizes the Add New command. When a user is clicked on this image a new empty row appears at the top of the grid and required fields become visible with the ❗ sign. (Figure 14)

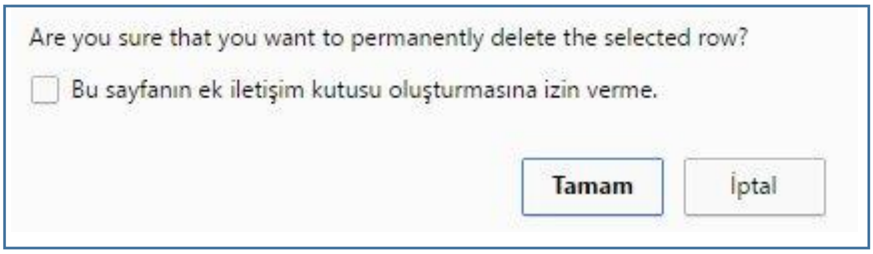


Powders | Finished Products | Production Capacities | Line Production Parameters | Line Setup Parameters

Powder Name	Inventory Cost (TL/Kg)	Subcontraction Cost (TL/Kg)	Powder Production Cost (TL/Kg)	⊕
❗	❗	❗	❗	⊗
Powder 1	10	10	2	⊗
Powder 2	10	10	2	⊗

Figure 14. Master data submenu

The Required Field warning disappears after user enter a value to each field. The ⊗ sign symbolizes the Delete Command. After the image is clicked a Confirm Dialog appears as in Figure 15. If user confirms his/her decision then the row is deleted. The Confirm Dialog is added in order to prevent loss of data mistakenly.

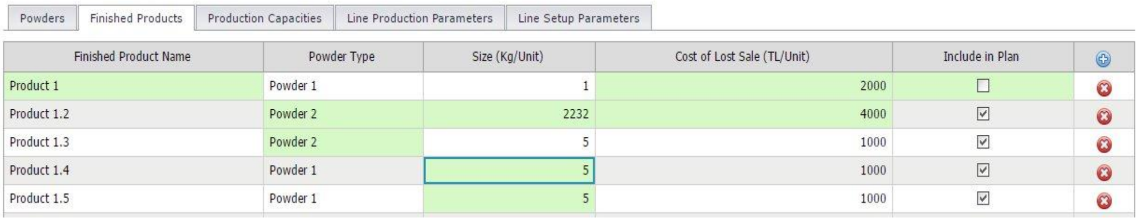


Are you sure that you want to permanently delete the selected row?

□ Bu sayfanın ek iletişim kutusu oluşturmasına izin verme.

Figure 15. Delete confirm dialogue

Data in ASPxGridView can be updated by just clicking the related cell. If the column is a text column like Name, any text is valid; or if the column type is ComboBox Column an option should be selected from the list. User can edit all cells at the same time. The edited cells become light green in order to be differentiated from the other cells. This property allow user to check his/her changes before saving as in Figure 16.



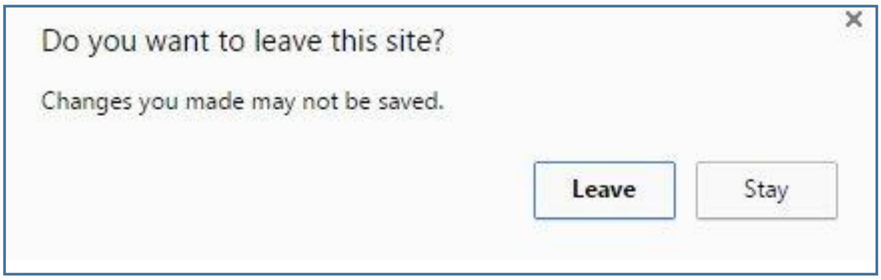
Powders | Finished Products | Production Capacities | Line Production Parameters | Line Setup Parameters

Finished Product Name	Powder Type	Size (Kg/Unit)	Cost of Lost Sale (TL/Unit)	Include in Plan	⊕
Product 1	Powder 1	1	2000	□	⊗
Product 1.2	Powder 2	2232	4000	☑	⊗
Product 1.3	Powder 2	5	1000	☑	⊗
Product 1.4	Powder 1	5	1000	☑	⊗
Product 1.5	Powder 1	5	1000	☑	⊗

Figure 16. Screen for master data editing

Furthermore, Validation Expressions are added when the need arises. As the name suggests, Validation Expression checks the validity of inputs. For example, Daily Capacity column in Production Capacities Page accepts only records with numbers. Texts or other characters are not allowed. Validation Expressions are important for data consistency. Besides, it reduces errors.

One last point about the ASPxGridView should also be emphasized. After adding, deleting or editing operations the Save changes button at the bottom right corner of the ASPxGridView have to be clicked. Otherwise, the database is not updated. If the user forgets to click Save changes button, a warning dialog appears before user leaves pages (Figure 17).



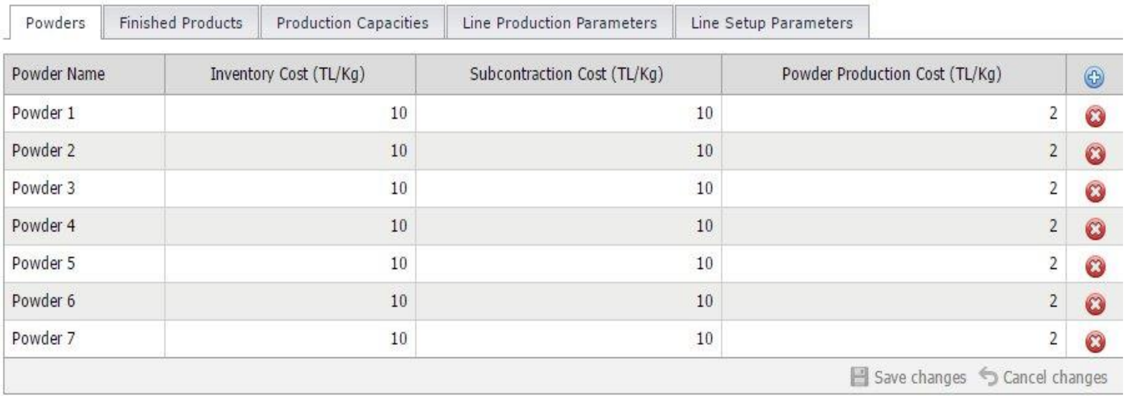
Do you want to leave this site?

Changes you made may not be saved.

Figure 17. Confirm dialog for leaving before saving changes

4.3.2.1 Powders page

Powders page includes powders’ name, inventory holding cost (kg), subcontracting cost (kg) and powder production cost (kg) information as it is seen in Figure 18. When a changes occurs related with powder, user should edit using this page.

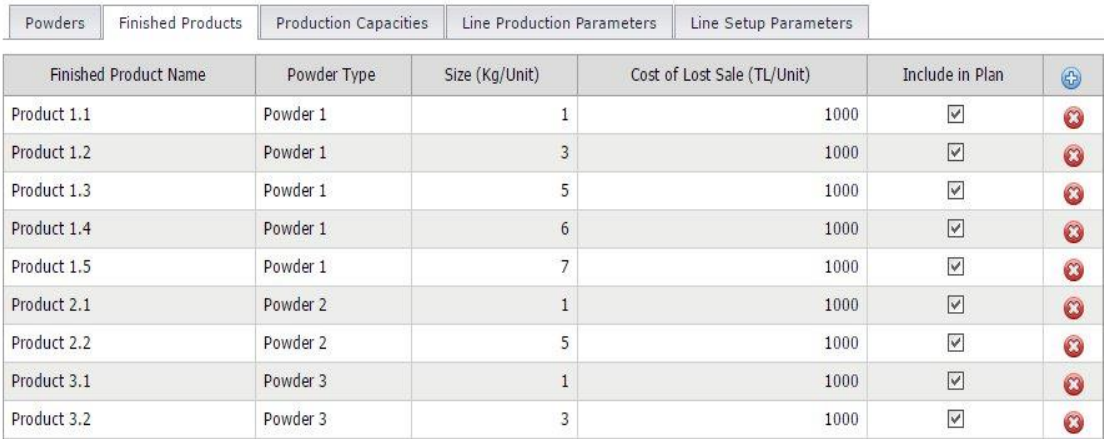


Powder Name	Inventory Cost (TL/Kg)	Subcontraction Cost (TL/Kg)	Powder Production Cost (TL/Kg)	⊕
Powder 1	10	10	2	⊗
Powder 2	10	10	2	⊗
Powder 3	10	10	2	⊗
Powder 4	10	10	2	⊗
Powder 5	10	10	2	⊗
Powder 6	10	10	2	⊗
Powder 7	10	10	2	⊗

Figure 18. Powders page

4.3.2.2 Finished products page

This page shows the details of finished products. In order to add a new finished product user should provide name, powder type, size (kg/unit), and cost of lost sale per unit. The image of finished product page is shown in Figure 19.



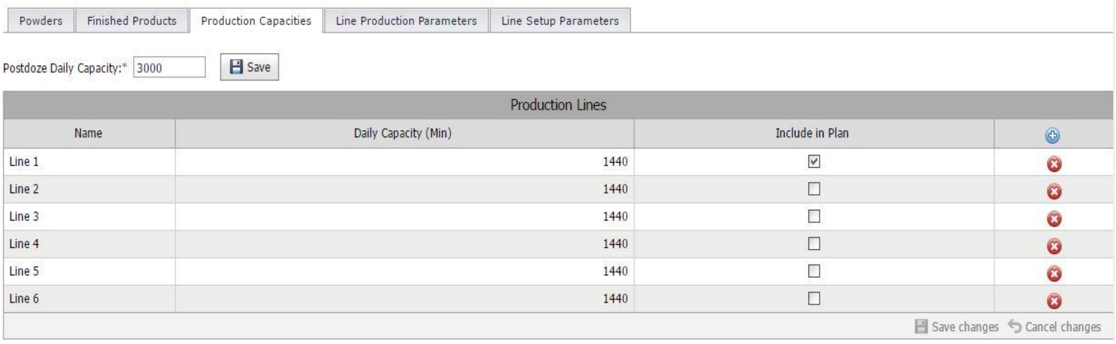
Finished Product Name	Powder Type	Size (Kg/Unit)	Cost of Lost Sale (TL/Unit)	Include in Plan	⊕
Product 1.1	Powder 1	1	1000	☑	⊗
Product 1.2	Powder 1	3	1000	☑	⊗
Product 1.3	Powder 1	5	1000	☑	⊗
Product 1.4	Powder 1	6	1000	☑	⊗
Product 1.5	Powder 1	7	1000	☑	⊗
Product 2.1	Powder 2	1	1000	☑	⊗
Product 2.2	Powder 2	5	1000	☑	⊗
Product 3.1	Powder 3	1	1000	☑	⊗
Product 3.2	Powder 3	3	1000	☑	⊗

Figure 19. Finished products page

Finished Products Page in Figure 19 has an Include in Plan column near to command column. Include in Plan column enable user to edit available finished products for the model. One may prefer to not to product all demanded finished products. In order to provide a flexible DSS this property is added.

4.3.2.4 Production capacities page

This page helps to edit data about two types of machines namely, postdoze and packing lines. Firstly, in ASPxGridView, there are three columns about production lines, Name, Daily Capacity and Include in Plan Column as in Figure 20.



Postdoze Daily Capacity:* 3000

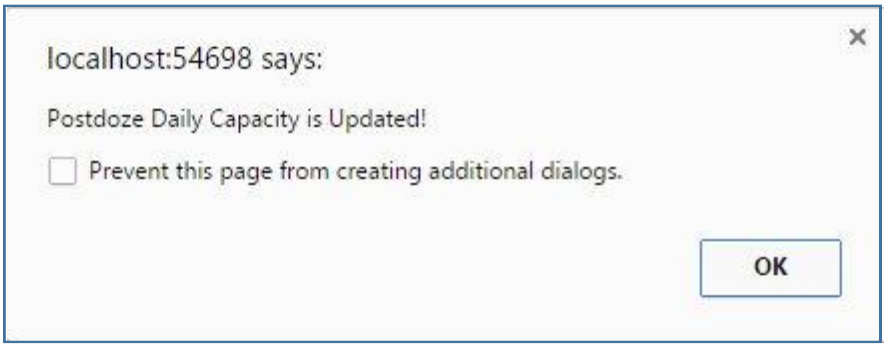
Production Lines

Name	Daily Capacity (Min)	Include in Plan
Line 1	1440	☑
Line 2	1440	□
Line 3	1440	□
Line 4	1440	□
Line 5	1440	□
Line 6	1440	□

Figure 20. Production capacities page

Name column represents the name of the production line. Daily Capacity column stores value for the available time in minutes. Normally, 1440 minutes available in a day but because of maintenance, cleaning or any other special reason a production line capacity may change. Lastly, similar to the Finished Products Page, the user may prefer not to include some production lines to plans.

The textbox above the ASPXGridView shows the daily capacity of postdoze machine. As stated before, this machine produces semi-finished products of detergents (powders). User can edit the daily capacity of postdoze machine by changing the data in textbox and clicking the Save button. Then the popup appears as in Figure 21.



localhost:54698 says:

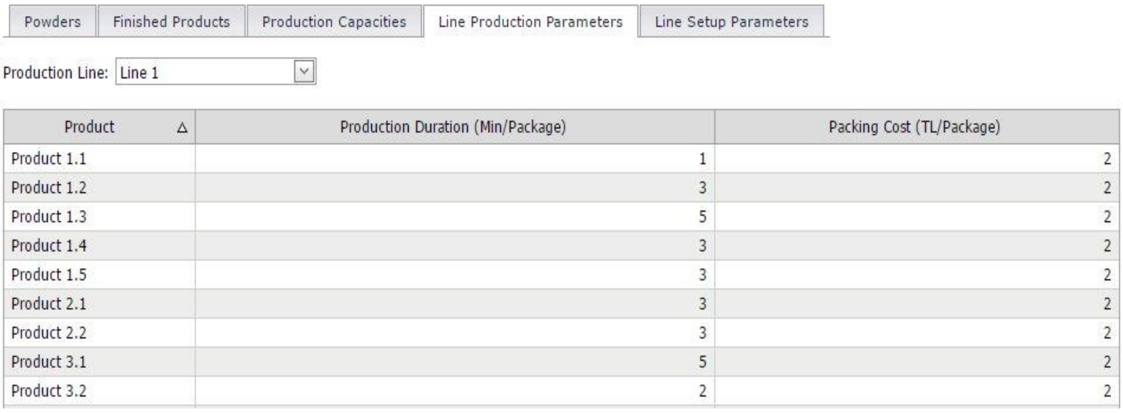
Postdoze Daily Capacity is Updated!

□ Prevent this page from creating additional dialogs.

Figure 21. Postdoze update popup

4.3.2.4 Line production parameters page

In the subject factory of this study, the packing costs and production durations of finished products change according to production lines. In this regard, the aim of Line Production Parameters Page is to enable user to edit production duration and packing cost values of each product on each line. Figure 22 shows the Line Production Parameters Page.



Production Line: Line 1

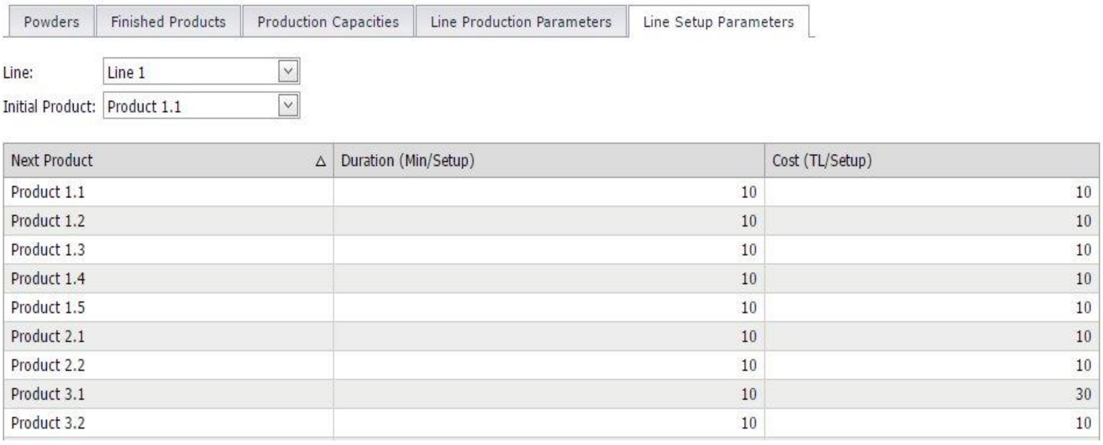
Product △	Production Duration (Min/Package)	Packing Cost (TL/Package)
Product 1.1	1	2
Product 1.2	3	2
Product 1.3	5	2
Product 1.4	3	2
Product 1.5	3	2
Product 2.1	3	2
Product 2.2	3	2
Product 3.1	5	2
Product 3.2	2	2

Figure 22. Line production parameters page

When user enters the page, s/he can only see a ComboBox. After choosing a production line, the ASPxGridView appears. Adding or deleting is not possible in this page because all products are automatically added to ASPxGridView. User can just edit production duration and packing cost columns.

4.3.2.5 Line setup parameters page

Line Setup Parameters Page is developer to insert setup durations and costs between the productions of two consecutive products on a selected line. As it is shown in the Figure 23, there are two combo box above the grid. The first ComboBox is to select production line and the second product is to select initial product. After those selections, user may edit the setup duration and cost information from initial product to next product on the related production line.



Line: Line 1

Initial Product: Product 1.1

Next Product △	Duration (Min/Setup)	Cost (TL/Setup)
Product 1.1	10	10
Product 1.2	10	10
Product 1.3	10	10
Product 1.4	10	10
Product 1.5	10	10
Product 2.1	10	10
Product 2.2	10	10
Product 3.1	10	30
Product 3.2	10	10

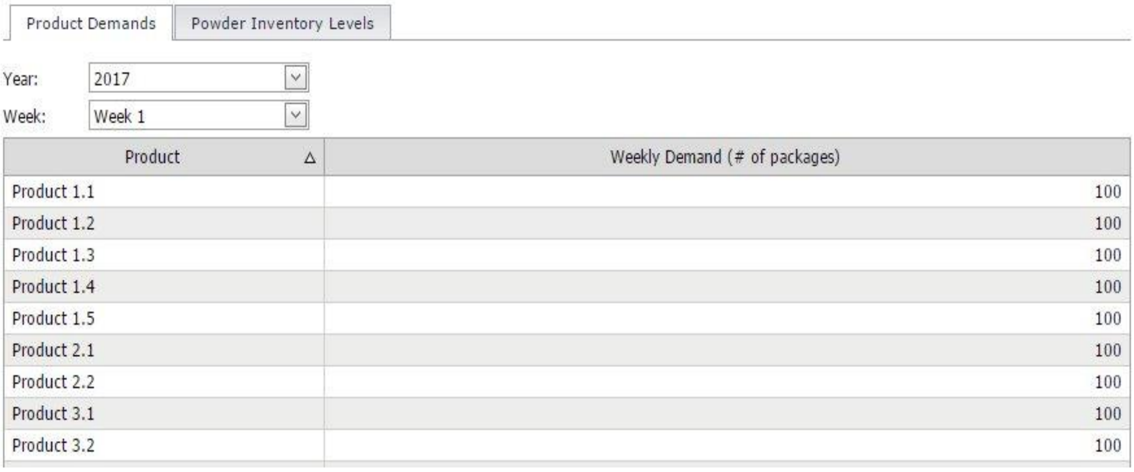
Figure 23. Line setup parameters page

4.3.3 Weekly data module

Weekly Data Module consists of two pages including Product Demands Page and Powder Inventory Level Page.

4.3.3.1 Product demands page

Product Demands Page is developed in order to manage Demand data of finished products. The appearance of the page can be seen in Figure 24.



Year: 2017

Week: Week 1

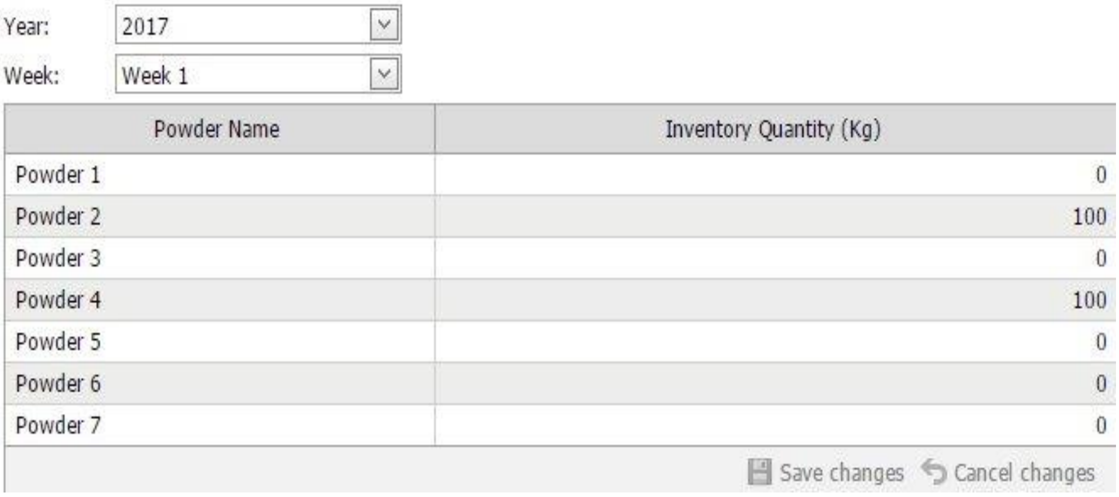
Product △	Weekly Demand (# of packages)
Product 1.1	100
Product 1.2	100
Product 1.3	100
Product 1.4	100
Product 1.5	100
Product 2.1	100
Product 2.2	100
Product 3.1	100
Product 3.2	100

Figure 24. Product demands page

There are two ComboBoxes at the top of the Demand Page. The first ComboBox is to select year and the second ComboBox is select the week. There is no Add New or Delete button in the Demands Page because when the year and week selected, one record for each finished product is added to database. The main aim of this automatic process is to ease use of GUI. If user adds a record for each product s/he may forget a product or s/he may add the same product a number of times. Also, automatic addition process is very time saving. User just needs to edit quantity column.

4.3.3.2 Powder inventory levels page

The aim of this page is to enable user to provide starting powder quantities to model. In subject factory of this study, finished product inventory is not hold. However, powder inventory may be held. In this regard, in order to help doing optimal production planning, user can insert the weekly initial values of each powder with the help of this page.



Year: 2017

Week: Week 1

Powder Name	Inventory Quantity (Kg)
Powder 1	0
Powder 2	100
Powder 3	0
Powder 4	100
Powder 5	0
Powder 6	0
Powder 7	0

Figure 25. Powder inventory levels page

4.3.4 Packing line schedule module

Packing Line Schedule Module is one of the most critical part of this DSS because this module not only present the actual schedules of production lines but also enable user to update those schedules. This module consists of 3 page. The first two page shows the schedules of each line in chart and tabular form respectively. And the third page presents daily schedule of all active lines.

4.3.4.1 Weekly schedule chart view page

When a user enters the Weekly Schedule Chart View Page s/he sees two combo boxes at the top of the page. If a week and a line is selected from these Comboboxes, the schedule of the related week and related line becomes visible as in Figure 26.

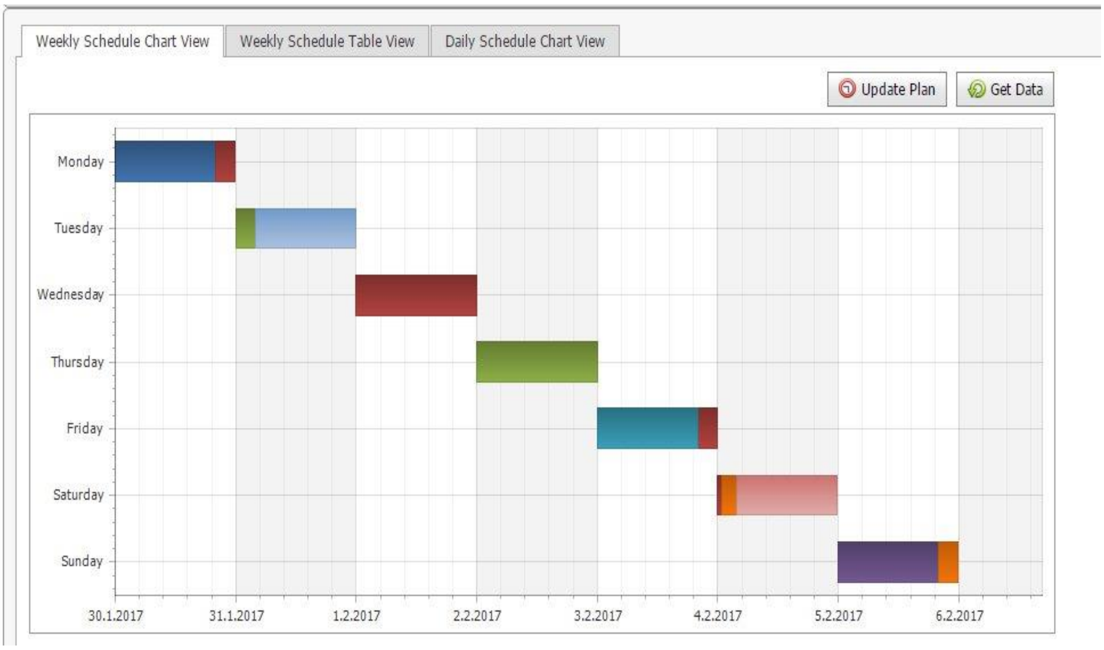


Figure 26. An output schedule

The main presentation tool for the schedule output is chart which is a special type of Gantt chart. Each production is added end to end with different colors. To put it other more clearly, in 30.01.2017 there are two different types of finished product productions. The name of the products become visible when user moves the cursor over the colored area (Figure 27).

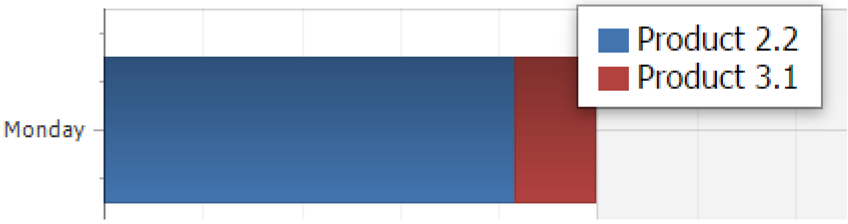
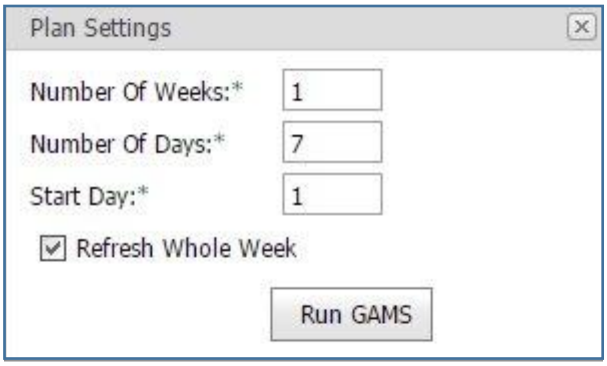


Figure 27. Product names when mouse hover over points

In order to update current plan or create new plan for other weeks, user should click Update Plan button. Then the popup on Figure 28 appears.



Plan Settings

Number Of Weeks:* 1

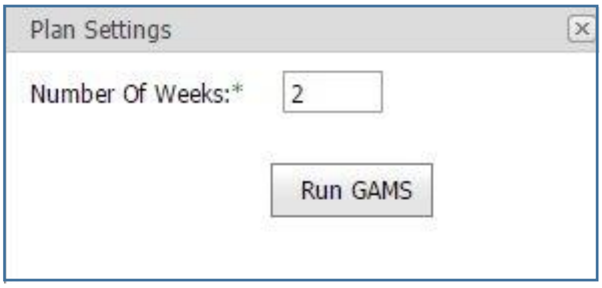
Number Of Days:* 7

Start Day:* 1

☑ Refresh Whole Week

Figure 28. Plan settings popup

Firstly, user should enter the number of weeks. If s/he wants to plan shorter than 1 week, s/he should also enter number of days and start day of plan. This fields are added because of the fact that user may needs to update plan at the middle of the week. If users wants to plan for more than one week, inserting the number of weeks is enough as in Figure 29.



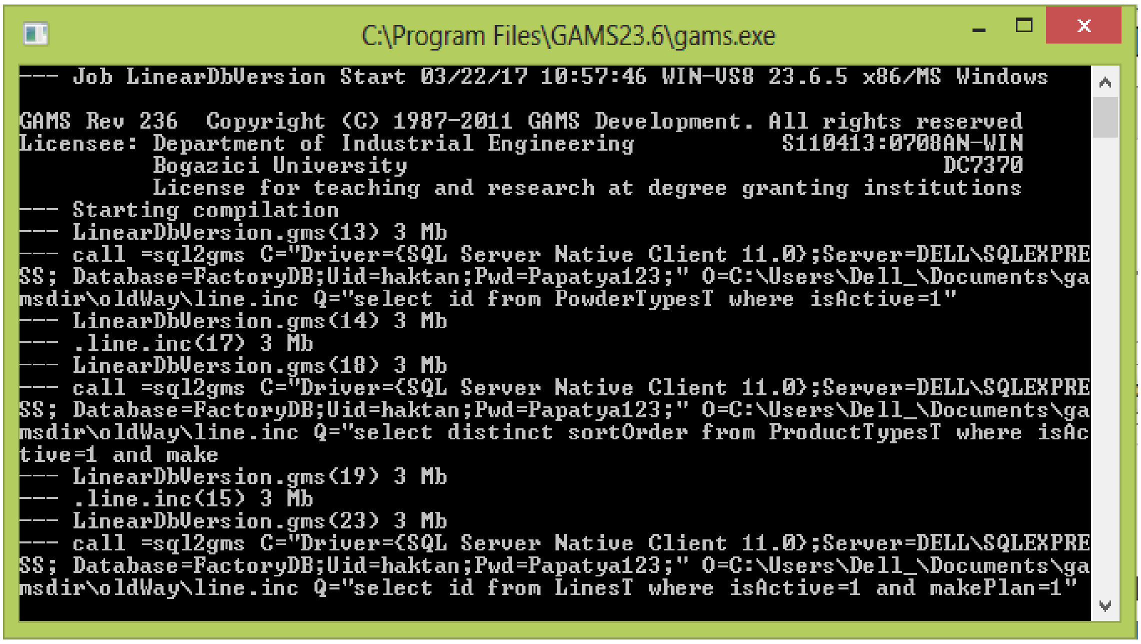
Plan Settings

Number Of Weeks:* 2

Figure 29. More than one week plan

Day values should be between 1 and 7 for the number of days and start day textboxes and 1 and 8 for the number of weeks textbox. Otherwise, system gives an error since solver limits are exceeded.

When user click the Run GAMS button with the plausible numbers, a command prompt appears as in Figure 30.

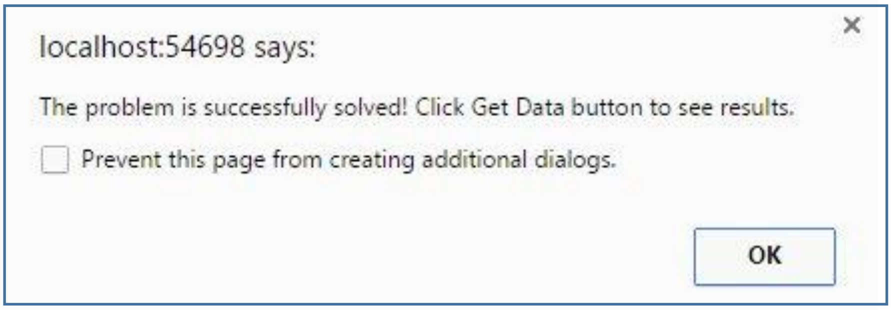


```
--- Job LinearDbVersion Start 03/22/17 10:57:46 WIN-VS8 23.6.5 x86/MS Windows

GAMS Rev 236  Copyright (C) 1987-2011 GAMS Development. All rights reserved
Licensee: Department of Industrial Engineering           S110413:0708AN-WIN
          Bogazici University                                        DC7370
          License for teaching and research at degree granting institutions
--- Starting compilation
--- LinearDbVersion.gms(13) 3 Mb
--- call =sql2gms C="Driver={SQL Server Native Client 11.0};Server=DELL\SQLEXPRE
SS; Database=FactoryDB;Uid=haktan;Pwd=Papatya123;" O=C:\Users\Dell_\Documents\ga
msdir\oldWay\line.inc Q="select id from PowderTypesT where isActive=1"
--- LinearDbVersion.gms(14) 3 Mb
--- .line.inc(17) 3 Mb
--- LinearDbVersion.gms(18) 3 Mb
--- call =sql2gms C="Driver={SQL Server Native Client 11.0};Server=DELL\SQLEXPRE
SS; Database=FactoryDB;Uid=haktan;Pwd=Papatya123;" O=C:\Users\Dell_\Documents\ga
msdir\oldWay\line.inc Q="select distinct sortOrder from ProductTypesT where isAc
tive=1 and make
--- LinearDbVersion.gms(19) 3 Mb
--- .line.inc(15) 3 Mb
--- LinearDbVersion.gms(23) 3 Mb
--- call =sql2gms C="Driver={SQL Server Native Client 11.0};Server=DELL\SQLEXPRE
SS; Database=FactoryDB;Uid=haktan;Pwd=Papatya123;" O=C:\Users\Dell_\Documents\ga
msdir\oldWay\line.inc Q="select id from LinesT where isActive=1 and makePlan=1"
```

Figure 30. Command prompt for GAMS solver

This prompt shows that GAMS Solver is running and trying to find a solution. Generally this process lasts around 1 minute. Then, the popup in Figure 31 appears.


localhost:54698 says:

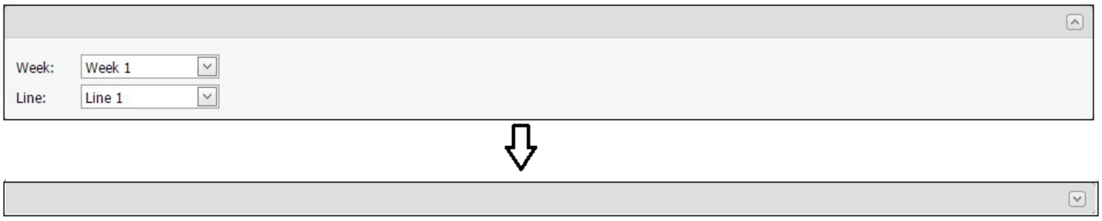
The problem is successfully solved! Click Get Data button to see results.

□ Prevent this page from creating additional dialogs.

Figure 31. Solver status popup

As a last step, user should click the Get Data button to see results. After this step, the chart refreshes itself and user can observe the optimal schedule. GAMS solver can be run many times if user needs.

It should be noted that the header panel that contains week and line comboboxes can be collapsed by clicking the gray part of the header as in Figure 32.



Week: Week 1

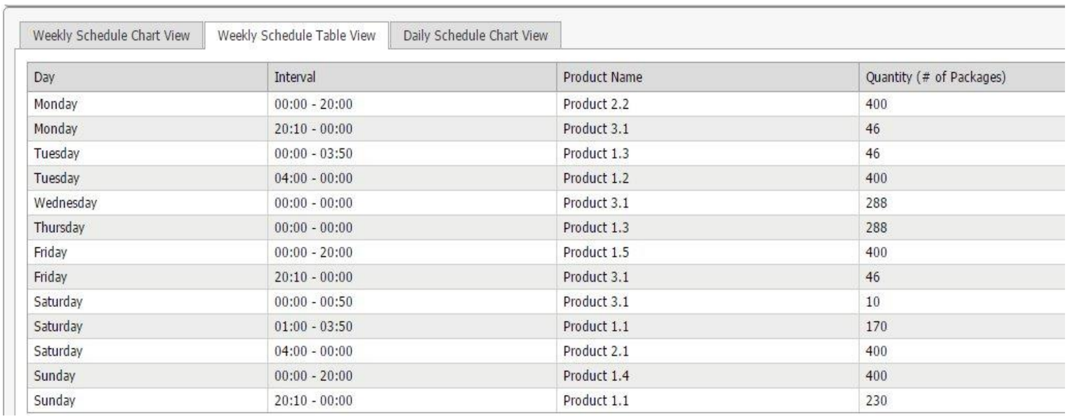
Line: Line 1

Figure 32. Header panel collapsing property

This property is added in order use the screen more effectively. User can collapse or open the header panel any time s/he needs by clicking the gray bar.

4.3.4.2 Weekly schedule table view

The schedule can also be seen in a tabular form by clicking Weekly Schedule Table View tab. The grid consists of 4 columns. The first column shows the day of the week. The second column shows the start and end times of productions. The third and fourth columns show product name and produced quantity respectively. A sample view of Weekly Schedule Table View is given in Figure 33.



Weekly Schedule Chart View
Weekly Schedule Table View
Daily Schedule Chart View

Day	Interval	Product Name	Quantity (# of Packages)
Monday	00:00 - 20:00	Product 2.2	400
Monday	20:10 - 00:00	Product 3.1	46
Tuesday	00:00 - 03:50	Product 1.3	46
Tuesday	04:00 - 00:00	Product 1.2	400
Wednesday	00:00 - 00:00	Product 3.1	288
Thursday	00:00 - 00:00	Product 1.3	288
Friday	00:00 - 20:00	Product 1.5	400
Friday	20:10 - 00:00	Product 3.1	46
Saturday	00:00 - 00:50	Product 3.1	10
Saturday	01:00 - 03:50	Product 1.1	170
Saturday	04:00 - 00:00	Product 2.1	400
Sunday	00:00 - 20:00	Product 1.4	400
Sunday	20:10 - 00:00	Product 1.1	230

Figure 33. Weekly schedule table view

4.3.4.3 Daily schedule chart view

Daily Schedule Chart has a similar view to the Weekly Schedule Chart but vertical axis shows the line names. Also, Daily Chart shows the daily plan of whole lines which are actively used in that week while Chart View shows the weekly plan of just one line. User should select a day in order to see chart. An example of Daily Schedule Chart is shown in Figure 37. As a last difference, product names are directly visible in this chart. Thus, a user can easily see the daily plan of whole production lines.

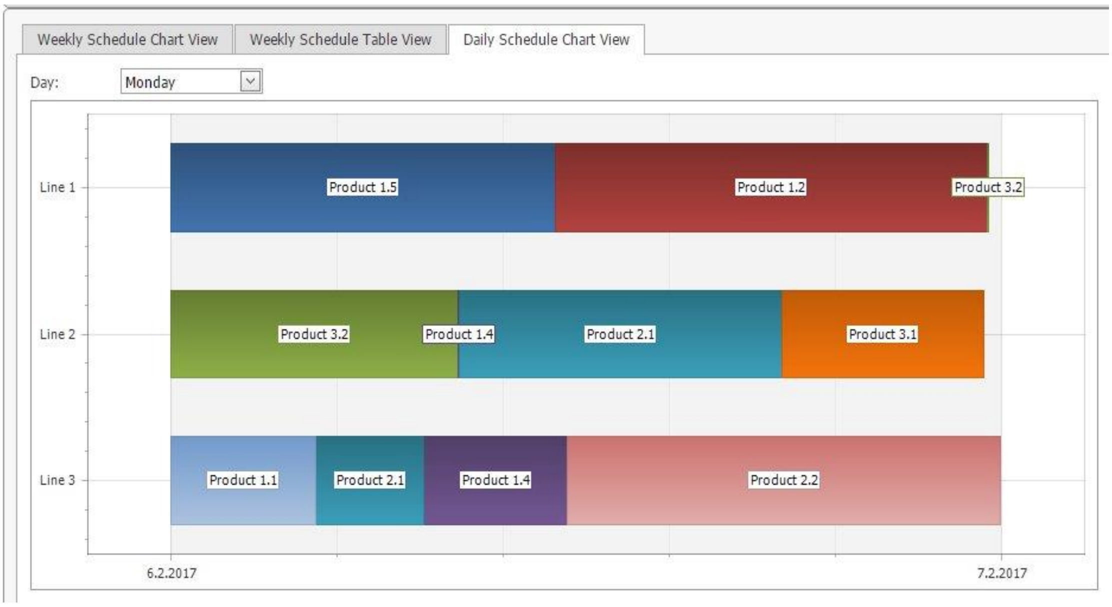


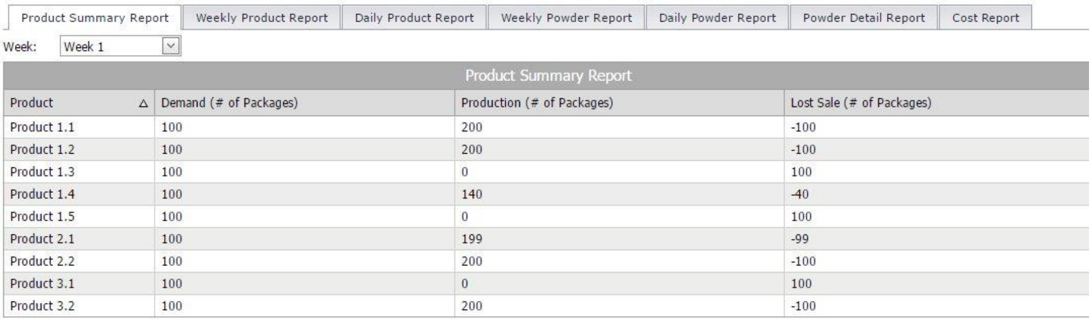
Figure 34. Daily schedule chart view example

4.3.5 Reports module

7 different report is prepared for the DSS.

4.3.5.1 Product summary report

Product Summary Report presents a basic summary of a selected week. Each row provides the product name, demand, production quantity and lost sale values in this table. An example of Product Summary Report is shown in Figure 35.



Week: Week 1

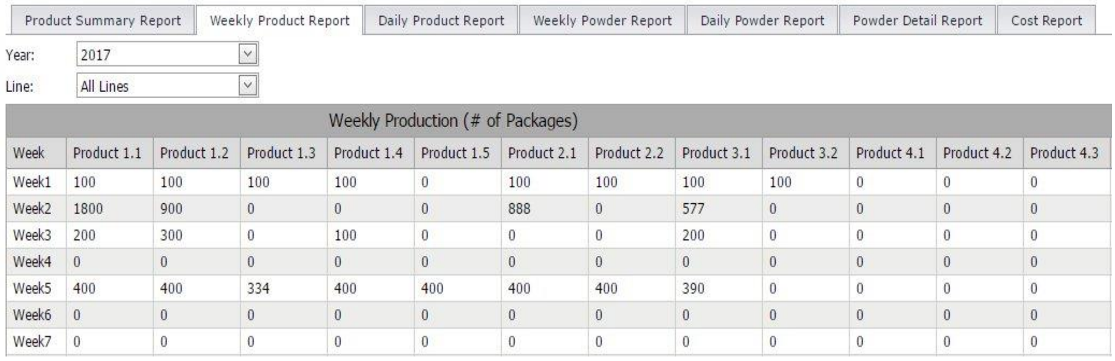
Product Summary Report

Product △	Demand (# of Packages)	Production (# of Packages)	Lost Sale (# of Packages)
Product 1.1	100	200	-100
Product 1.2	100	200	-100
Product 1.3	100	0	100
Product 1.4	100	140	-40
Product 1.5	100	0	100
Product 2.1	100	199	-99
Product 2.2	100	200	-100
Product 3.1	100	0	100
Product 3.2	100	200	-100

Figure 35. Product summary report

4.3.5.2 Weekly product report

Product Report provides information about the weekly distribution of whole finished products. A user can easily observe the whole year’s production. Also, s/he can filter results according to the line in case of necessities with the help of the combobox that is positioned at the top of the page. Figure 36 shows Product Report.



Year: 2017

Line: All Lines

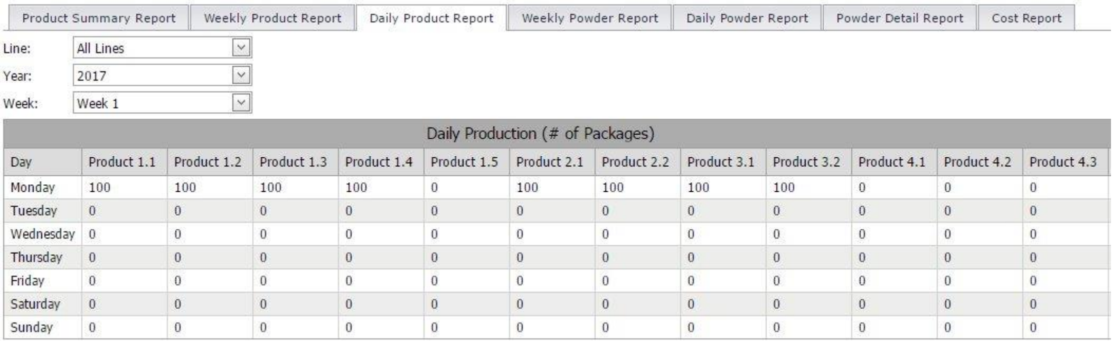
Weekly Production (# of Packages)

Week	Product 1.1	Product 1.2	Product 1.3	Product 1.4	Product 1.5	Product 2.1	Product 2.2	Product 3.1	Product 3.2	Product 4.1	Product 4.2	Product 4.3
Week1	100	100	100	100	0	100	100	100	100	0	0	0
Week2	1800	900	0	0	0	888	0	577	0	0	0	0
Week3	200	300	0	100	0	0	0	200	0	0	0	0
Week4	0	0	0	0	0	0	0	0	0	0	0	0
Week5	400	400	334	400	400	400	400	390	0	0	0	0
Week6	0	0	0	0	0	0	0	0	0	0	0	0
Week7	0	0	0	0	0	0	0	0	0	0	0	0

Figure 36. Weekly product report

4.3.5.3 Daily product report

Daily Product Report is a special version of Product Report. Daily distribution of finished product productions can be seen from this report. User just need to select a week. (Figure 37)



Line: All Lines
Year: 2017
Week: Week 1

Daily Production (# of Packages)

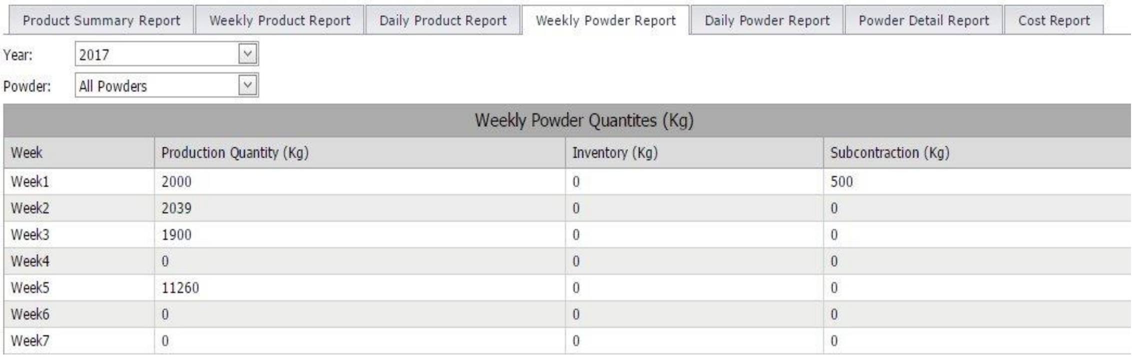
Day	Product 1.1	Product 1.2	Product 1.3	Product 1.4	Product 1.5	Product 2.1	Product 2.2	Product 3.1	Product 3.2	Product 4.1	Product 4.2	Product 4.3
Monday	100	100	100	100	0	100	100	100	100	0	0	0
Tuesday	0	0	0	0	0	0	0	0	0	0	0	0
Wednesday	0	0	0	0	0	0	0	0	0	0	0	0
Thursday	0	0	0	0	0	0	0	0	0	0	0	0
Friday	0	0	0	0	0	0	0	0	0	0	0	0
Saturday	0	0	0	0	0	0	0	0	0	0	0	0
Sunday	0	0	0	0	0	0	0	0	0	0	0	0

Figure 37. Daily product report

It should be noted that daily productions can be filtered according to lines if needed. The line combobox at the top of the page shows the selected line. The default value of this combobox named all lines shows the production of all lines.

4.3.5.4 Weekly powder report

Weekly Powder Report indicates the produced, inventoried and subcontracted quantities for whole week. User can filter the results based on powder type if s/he needs. An example of Powder Report is shown in Figure 38.



Year: 2017
Powder: All Powders

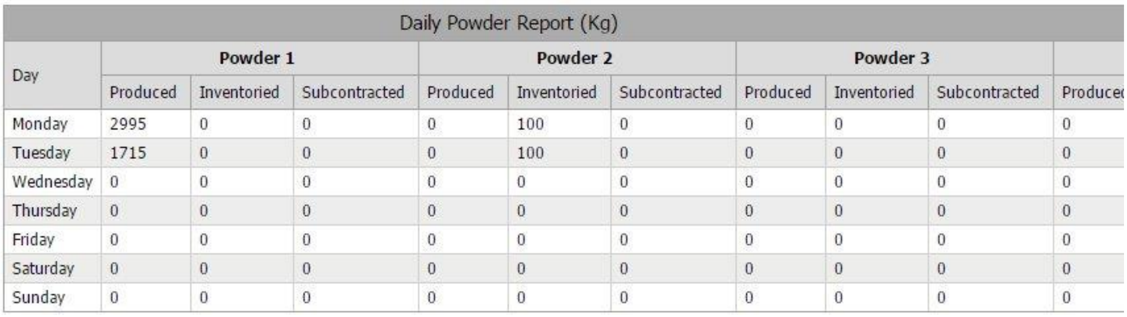
Weekly Powder Quantites (Kg)

Week	Production Quantity (Kg)	Inventory (Kg)	Subcontraction (Kg)
Week1	2000	0	500
Week2	2039	0	0
Week3	1900	0	0
Week4	0	0	0
Week5	11260	0	0
Week6	0	0	0
Week7	0	0	0

Figure 38. Weekly powder report

4.3.5.5 Daily powder report

Daily Powder Report shows the daily distribution of produced, inventoried and subcontracted powder quantities for the selected week.



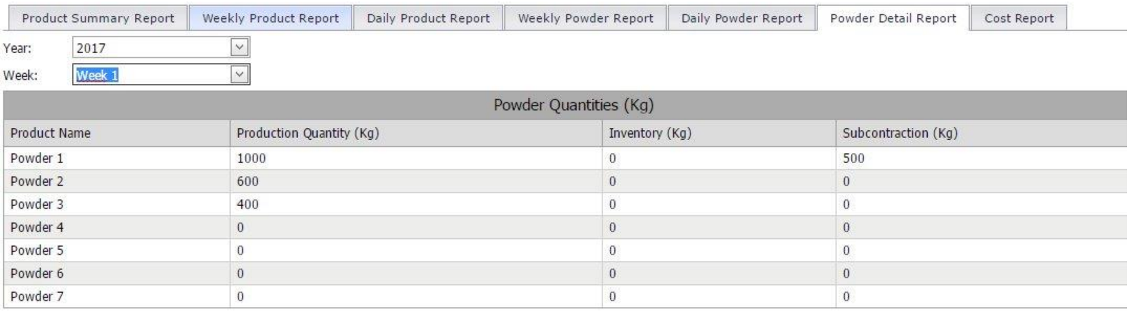
Daily Powder Report (Kg)

Day	Powder 1			Powder 2			Powder 3			
	Produced	Inventoried	Subcontracted	Produced	Inventoried	Subcontracted	Produced	Inventoried	Subcontracted	Produce
Monday	2995	0	0	0	100	0	0	0	0	0
Tuesday	1715	0	0	0	100	0	0	0	0	0
Wednesday	0	0	0	0	0	0	0	0	0	0
Thursday	0	0	0	0	0	0	0	0	0	0
Friday	0	0	0	0	0	0	0	0	0	0
Saturday	0	0	0	0	0	0	0	0	0	0
Sunday	0	0	0	0	0	0	0	0	0	0

Figure 39. Daily powder report

4.3.5.6 Powder detail report

Powder Detail Report shows the produced, inventoried and subcontracted quantities of different powder types in a week. User should select the week number in order to see the report. (Figure 40)



Year: 2017

Week: Week 1

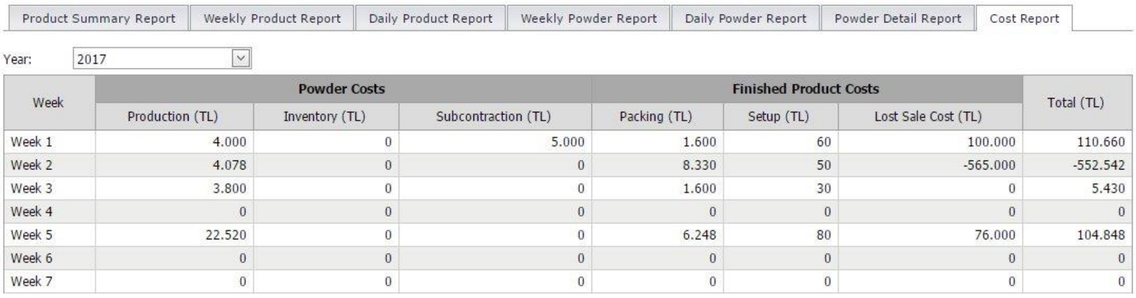
Powder Quantities (Kg)

Product Name	Production Quantity (Kg)	Inventory (Kg)	Subcontraction (Kg)
Powder 1	1000	0	500
Powder 2	600	0	0
Powder 3	400	0	0
Powder 4	0	0	0
Powder 5	0	0	0
Powder 6	0	0	0
Powder 7	0	0	0

Figure 40. Powder detail report

4.3.5.7 Cost report

Cost report indicates the costs of each week plans. With the help of this report, a user can easily observe the distributions of different cost types.



Year: 2017

Week	Powder Costs			Finished Product Costs			Total (TL)
	Production (TL)	Inventory (TL)	Subcontraction (TL)	Packing (TL)	Setup (TL)	Lost Sale Cost (TL)	
Week 1	4.000	0	5.000	1.600	60	100.000	110.660
Week 2	4.078	0	0	8.330	50	-565.000	-552.542
Week 3	3.800	0	0	1.600	30	0	5.430
Week 4	0	0	0	0	0	0	0
Week 5	22.520	0	0	6.248	80	76.000	104.848
Week 6	0	0	0	0	0	0	0
Week 7	0	0	0	0	0	0	0

Figure 41. Cost report

CHAPTER 5

CONCLUSION

The inspiration of this thesis study comes from the smart manufacturing concept that is introduced with the 4th Industry 4.0 targets. Today, companies are trying to combine planning and manufacturing much faster and more flexibility than before by the use of IoT. In this digitalized environment, manufacturing systems are expected to respond planned or unplanned events much faster, with less errors and from different mediums such as tablets, mobile devices. This requires that DSSs should be designed with much more efficiency, effectiveness and flexibility.

In this thesis study, a DSS is developed in order to enhance the quality of production planning and scheduling decisions of a detergent factory. The DSS provides an effective decision making environment by decreasing the number of errors in decision making and increasing the quality of decision making environment. In order reduce number of errors validation expressions, confirm boxes and warnings are widely used. Moreover, directive notes are added in order to help user in using the system free of error. After inserting the master data, the use of system requires very little time. Thus, decision making process becomes faster and more efficient. User can reach optimal result without much effort. In addition to that, updating existing plans or what-if analysis are very easy to apply. Since production planning and scheduling are handled together in this DSS, more realistic plans can be reached.

It should also be noted that the DSS provides an efficient and effective way to manage data. All information can be reached easily and updated without effecting the database. Also, previous plans are stored and can be used easily. Moreover, the reports provide very effective feedback by showing time and cost analysis of actual or previous schedules. DSS also provide managerial implications with the help of reports.

Finally, a flexible environment is provided with the help of graphical user interface. To illustrate, user can update plans at the middle of week without too much any extra effort. Thus, the plans are not very sensitive to unexpected changes like machine breakdowns, raw material deficiencies or demand changes. If a machine breakdown occurs, user can reschedule in minutes by just removing the problematic machine from the model or by updating its capacity with value obtained from IoT environment. The main advantage of this DSS is the quick response ability to changes that is gathered over IoT.

For future studies, an integration with the production lines may be developed. With the help of this connection, the actual values can be shown in the DSS by comparing the planned values. Also, the problems with the machines automatically may be reported to DSS and automatic rescheduling may be possible. In addition, the plans may be automatically sent to machines in an IoT environment.

Another future study may include heuristic methodologies because when the problem size gets bigger, the needed time for the optimal solution gets very high. Simulated annealing and Tabu search algorithms may be tested.

In another future study, user roles may be included. Thus, the distributions of roles may be clearer in DSS. For example, just one responsible user can edit master data and just authorized users may see reports. Those kinds of properties may reduce errors and enhance the effectiveness of DSS.

Finally, a future work may be the mobile application development. Although, responsive designs enable user to reach via mobile phone. Mobile applications may be more efficient.

APPENDIX

GAMS MODEL

```
sets

p "position"        /1 * 4/,

o "orderCount"      /0 * 4/;


set i "powders"  /

$call =sql2gms C="Driver={SQL Server Native Client
11.0};Server=DELL\SQLEXPRESS;
Database=FactoryDB;Uid=haktan;Pwd=Papatya123;"
O=C:\Users\Dell_\Documents\gamsdir\oldWay\line.inc Q="select id from
PowderTypesT where isActive=1"

$include C:\Users\Dell_\Documents\gamsdir\oldWay\line.inc

/;


set j "type"  /

$call =sql2gms C="Driver={SQL Server Native Client
11.0};Server=DELL\SQLEXPRESS;
Database=FactoryDB;Uid=haktan;Pwd=Papatya123;"
O=C:\Users\Dell_\Documents\gamsdir\oldWay\line.inc Q="select
distinct sortOrder from ProductTypesT where isActive=1 and
makePlan=1"

$include C:\Users\Dell_\Documents\gamsdir\oldWay\line.inc

/;


set k "line"   /

$call =sql2gms C="Driver={SQL Server Native Client
11.0};Server=DELL\SQLEXPRESS;
Database=FactoryDB;Uid=haktan;Pwd=Papatya123;"
O=C:\Users\Dell_\Documents\gamsdir\oldWay\line.inc Q="select id from
LinesT where isActive=1 and makePlan=1"

$include C:\Users\Dell_\Documents\gamsdir\oldWay\line.inc

/;


set t "day"  /

$call =sql2gms C="Driver={SQL Server Native Client
11.0};Server=DELL\SQLEXPRESS;
Database=FactoryDB;Uid=haktan;Pwd=Papatya123;"
O=C:\Users\Dell_\Documents\gamsdir\oldWay\line.inc
Q="SelectCustomDays"
```

```
$include C:\Users\Dell_\Documents\gamsdir\oldWay\line.inc

/;


set w "week"  /

$call =sql2gms C="Driver={SQL Server Native Client
11.0};Server=DELL\SQLEXPRESS;
Database=FactoryDB;Uid=haktan;Pwd=Papatya123;"
O=C:\Users\Dell_\Documents\gamsdir\oldWay\line.inc Q="SelectWeeks"

$include C:\Users\Dell_\Documents\gamsdir\oldWay\line.inc

/;


alias (i,iP);

alias (j,jP);


POSITIVE VARIABLES Inv(i,t,w), S(i,t,w), X(i,t,w),
Y(i,j,iP,jP,p,k,t,w) ;

VARIABLE Z, A(k,t,w);

Binary Variables  Pi(o,k,t,w), M(i,j,k,t,w), E(i,j,p,k,t,w);

Integer Variables  Q(i,j,k,t,w);


Y.up(i,j,iP,jP,p,k,t,w)   = 1;

Y.lo(i,j,iP,jP,p,k,t,w)   = 0;


A.lo(k,t,w) = 0;

A.up(k,t,w) = 1000;


Q.up(i,j,k,t,w) = 10000;

Q.lo(i,j,k,t,w) = 0;


PARAMETERS d(i,j,w), f(i,j,iP,jP,k), size(i,j), duration(i,j,k),
cSub(i), cLost(i,j), cPack(i,j,k), cPow(i), cSetup(i,j,iP,jP,k),

dailyCap(k), PostdozeCap, InvStart(i);


parameter cInv(i) /

$call =sql2gms C="Driver={SQL Server Native Client
11.0};Server=DELL\SQLEXPRESS;
Database=FactoryDB;Uid=haktan;Pwd=Papatya123;"
O=C:\Users\Dell_\Documents\gamsdir\oldWay\inv.inc Q="select id,
inventoryCost from PowderTypesT where isActive=1"

$include C:\Users\Dell_\Documents\gamsdir\oldWay\inv.inc
```

```
/;


parameter cSub(i) /

$call =sql2gms C="Driver={SQL Server Native Client
11.0};Server=DELL\SQLEXPRESS;
Database=FactoryDB;Uid=haktan;Pwd=Papatya123;"
O=C:\Users\Dell_\Documents\gamsdir\oldWay\sub.inc Q="select id,
subContractCost from PowderTypesT where isActive=1"

$include C:\Users\Dell_\Documents\gamsdir\oldWay\sub.inc

/;


parameter cPow(i) /

$call =sql2gms C="Driver={SQL Server Native Client
11.0};Server=DELL\SQLEXPRESS;
Database=FactoryDB;Uid=haktan;Pwd=Papatya123;"
O=C:\Users\Dell_\Documents\gamsdir\oldWay\pow.inc Q="select id,
powderProductionCost from PowderTypesT where isActive=1"

$include C:\Users\Dell_\Documents\gamsdir\oldWay\pow.inc

/;


parameter cLost(i, j) /

$call =sql2gms C="Driver={SQL Server Native Client
11.0};Server=DELL\SQLEXPRESS;
Database=FactoryDB;Uid=haktan;Pwd=Papatya123;"
O=C:\Users\Dell_\Documents\gamsdir\oldWay\lost.inc Q="select
powderTypeId, sortOrder, lostCost from ProductTypesT where
isActive=1 and makePlan=1"

$include C:\Users\Dell_\Documents\gamsdir\oldWay\lost.inc

/;


parameter size(i, j) /

$call =sql2gms C="Driver={SQL Server Native Client
11.0};Server=DELL\SQLEXPRESS;
Database=FactoryDB;Uid=haktan;Pwd=Papatya123;"
O=C:\Users\Dell_\Documents\gamsdir\oldWay\size.inc Q="select
powderTypeId, sortOrder, size from ProductTypesT where isActive=1
and makePlan=1"

$include C:\Users\Dell_\Documents\gamsdir\oldWay\size.inc

/;


parameter duration(i,j,k) /

$call =sql2gms C="Driver={SQL Server Native Client
11.0};Server=DELL\SQLEXPRESS;
Database=FactoryDB;Uid=haktan;Pwd=Papatya123;"
O=C:\Users\Dell_\Documents\gamsdir\oldWay\duration.inc Q="select
```

```
powderTypeId, sortOrder, lineSortOrder, duration from
ProductionCostsV where isActive=1 and makePlan=1"

$include C:\Users\Dell_\Documents\gamsdir\oldWay\duration.inc

/;


parameter d(i,j,w) /

$call =sql2gms C="Driver={SQL Server Native Client
11.0};Server=DELL\SQLEXPRESS;
Database=FactoryDB;Uid=haktan;Pwd=Papatya123;"
O=C:\Users\Dell_\Documents\gamsdir\oldWay\demand.inc
Q="SelectDemands"

$include C:\Users\Dell_\Documents\gamsdir\oldWay\demand.inc

/;


parameter cPack(i,j,k) /

$call =sql2gms C="Driver={SQL Server Native Client
11.0};Server=DELL\SQLEXPRESS;
Database=FactoryDB;Uid=haktan;Pwd=Papatya123;"
O=C:\Users\Dell_\Documents\gamsdir\oldWay\pack.inc Q="select
powderTypeId, sortOrder, lineId, cost from ProductionCostsV where
isActive=1 and makePlan=1  and prodMakePlan=1"

$include C:\Users\Dell_\Documents\gamsdir\oldWay\pack.inc

/;


parameter InvStart(i) /

$call =sql2gms C="Driver={SQL Server Native Client
11.0};Server=DELL\SQLEXPRESS;
Database=FactoryDB;Uid=haktan;Pwd=Papatya123;"
O=C:\Users\Dell_\Documents\gamsdir\oldWay\InvStart.inc
Q="SelectInventoryLevels"

$include C:\Users\Dell_\Documents\gamsdir\oldWay\InvStart.inc

/;


parameter dailyCap(k) /

$call =sql2gms C="Driver={SQL Server Native Client
11.0};Server=DELL\SQLEXPRESS;
Database=FactoryDB;Uid=haktan;Pwd=Papatya123;"
O=C:\Users\Dell_\Documents\gamsdir\oldWay\cap.inc Q="select id,
dailyCapacity from LinesT where isActive=1 and makePlan=1"

$include C:\Users\Dell_\Documents\gamsdir\oldWay\cap.inc

/;


parameter PostdozeCap /
```

```
$call =sql2gms C="Driver={SQL Server Native Client
11.0};Server=DELL\SQLEXPRESS;
Database=FactoryDB;Uid=haktan;Pwd=Papatya123;"
O=C:\Users\Dell_\Documents\gamsdir\oldWay\postCap.inc Q="Select
postdozeCapacity from WeeklyPlanSettingsT"

$include C:\Users\Dell_\Documents\gamsdir\oldWay\postCap.inc

/;


parameter cSetup(i,j,iP,jP,k) /

$call =sql2gms C="Driver={SQL Server Native Client
11.0};Server=DELL\SQLEXPRESS;
Database=FactoryDB;Uid=haktan;Pwd=Papatya123;"
O=C:\Users\Dell_\Documents\gamsdir\oldWay\setup.inc Q="select
powderTypeId, sortOrder, nextPowderTypeId, nextSortOrder, lineId,
cost from SetupCostsV where isActive=1 and nextIsActive=1 and
makePlan=1 and prodMakePlan=1 and nextProdMakePlan=1"

$include C:\Users\Dell_\Documents\gamsdir\oldWay\setup.inc

/;


parameter f(i,j,iP,jP,k) /

$call =sql2gms C="Driver={SQL Server Native Client
11.0};Server=DELL\SQLEXPRESS;
Database=FactoryDB;Uid=haktan;Pwd=Papatya123;"
O=C:\Users\Dell_\Documents\gamsdir\oldWay\f.inc Q="select
powderTypeId, sortOrder, nextPowderTypeId, nextSortOrder, lineId,
duration from SetupCostsV where isActive=1 and nextIsActive=1 and
makePlan=1 and prodMakePlan=1 and nextProdMakePlan=1"

$include C:\Users\Dell_\Documents\gamsdir\oldWay\f.inc

/;




EQUATIONS  CON1A, CON1B, CON2, CON3, CON4, CON5, CON6, CON7, CON8,
CON9, CON10A, CON10B, CON11A, CON11B, CON12A, CON12B, CON13A,
CON13B, CON14, CON15, CON16, CON17, CON18, OBJ;


CON1A(i,t,w)$(t.val=1)..

      sum((j,k), Q(i,j,k,t,w)*size(i,j))  =e=   X(i,t,w) -
      Inv(i,t,w) + InvStart(i) + S(i,t,w);


CON1B(i,t,w)$(t.val>1)..

      sum((j,k), Q(i,j,k,t,w)*size(i,j)) =e=   X(i,t,w) - Inv(i,t,w)
      + Inv(i,t-1,w) + S(i,t,w);


CON2(i,t,w)..
```

```
      Inv(i,'7',w) =e= Inv(i, '1', w+1);


CON3(i,j,w)..

      sum((t,k), Q(i,j,k,t,w)) =l= d(i,j,w);


CON4(k,t,w)..

      sum((i,j), Q(i,j,k,t,w) * duration(i,j,k))  + A(k,t,w)   =l=
      dailyCap(k);


CON5(t,w)..

      sum((i), X(i,t,w)) =l= PostdozeCap;


CON6(k,t,w)..

      A(k,t,w) =e=  sum((i,j,iP,jP), f(i,j,iP,jP,k) * sum(p,
      Y(i,j,iP,jP,p,k,t,w)));


CON7(i,j,k,t,w)..

      M(i,j,k,t,w) =l= Q(i,j,k,t,w);


CON8(i,j,k,t,w)..

      d(i,j,w) * M(i,j,k,t,w) =g= Q(i,j,k,t,w);


CON9(k,t,w)..

      Pi('0',k,t,w) + Pi('1',k,t,w) + Pi('2',k,t,w) + Pi('3',k,t,w)
      + Pi('4',k,t,w) =e= 1;


CON10A(i,j,k,t,w)..

      E(i,j,'1',k,t,w) - M(i,j,k,t,w) =l= 1-Pi('1',k,t,w);


CON10B(i,j,k,t,w)..

      -E(i,j,'1',k,t,w) + M(i,j,k,t,w) =l= 1-Pi('1',k,t,w);


CON11A(i,j,k,t,w)..

      E(i,j,'1',k,t,w) + E(i,j,'2',k,t,w) - M(i,j,k,t,w) =l= 2 * (1-
      Pi('2',k,t,w));


CON11B(i,j,k,t,w)..
```

```
      -E(i,j,'1',k,t,w) - E(i,j,'2',k,t,w) + M(i,j,k,t,w) =l= 2 *
      (1-Pi('2',k,t,w));


CON12A(i,j,k,t,w)..

      E(i,j,'1',k,t,w) + E(i,j,'2',k,t,w) + E(i,j,'3',k,t,w) -
      M(i,j,k,t,w) =l= 3 * (1-Pi('3',k,t,w));


CON12B(i,j,k,t,w)..

      -E(i,j,'1',k,t,w) - E(i,j,'2',k,t,w) - E(i,j,'3',k,t,w) +
      M(i,j,k,t,w) =l= 3 * (1-Pi('3',k,t,w));


CON13A(i,j,k,t,w)..

      E(i,j,'1',k,t,w) + E(i,j,'2',k,t,w) + E(i,j,'3',k,t,w) +
      E(i,j,'4',k,t,w) - M(i,j,k,t,w) =l= 4 * (1-Pi('4',k,t,w));


CON13B(i,j,k,t,w)..

      -E(i,j,'1',k,t,w) - E(i,j,'2',k,t,w) - E(i,j,'3',k,t,w) -
      E(i,j,'4',k,t,w) + M(i,j,k,t,w) =l= 4 * (1-Pi('4',k,t,w));


CON14(k,t,w)..

      sum((i,j), E(i,j,'1',k,t,w)) =e= 1 - Pi('0',k,t,w);


CON15(k,t,w)..

      sum((i,j), E(i,j,'2',k,t,w)) =e= 1 - Pi('0',k,t,w) -
Pi('1',k,t,w);


CON16(k,t,w)..

      sum((i,j), E(i,j,'3',k,t,w)) =e= 1 - Pi('0',k,t,w) -
      Pi('1',k,t,w) - Pi('2',k,t,w);


CON17(k,t,w)..

      sum((i,j), E(i,j,'4',k,t,w)) =e= 1 - Pi('0',k,t,w) -
      Pi('1',k,t,w) - Pi('2',k,t,w) - Pi('3',k,t,w);


CON18(p,k,t,w,i,j,iP,jP)$(p.val<=3)..

      Y(i,j,iP,jP,p,k,t,w)=g= E(i,j,p,k,t,w) + E(iP,jP,p+1,k,t,w) -
1 ;


OBJ..
```

```
      Z =e=         Sum((w, i), Sum(t, cSub(i) * S(i,t,w) + cInv(i)
      * Inv(i,t,w) + cPow(i) *  X(i,t,w) + sum((j,k), cPack(i,j,k) *
      Q(i,j,k,t,w))) + Sum(j, cLost(i,j) * (d(i,j,w) - sum((t, k),
      Q(i,j,k,t,w))))) + Sum((i,j,iP,jP,p,k,t,w),
      cSetup(i,j,iP,jP,k)* Y(i,j,iP,jP,p,k,t,w));



MODEL MyOptimizationModal / ALL / ;

MyOptimizationModal.OptFile = 1;

SOLVE MyOptimizationModal MINIMIZING Z USING MIP;


execute_unload
'C:\Users\Dell_\Documents\gamsdir\projdir\readData.gdx',E,Q,X,Inv,S;

execute 'gdxviewer.exe
i=C:\Users\Dell_\Documents\gamsdir\projdir\readData.gdx
csv=C:\Users\Dell_\Documents\gamsdir\projdir\resultsEta.csv id=E';

execute 'gdxviewer.exe
i=C:\Users\Dell_\Documents\gamsdir\projdir\readData.gdx
csv=C:\Users\Dell_\Documents\gamsdir\projdir\resultsQuantity.csv
id=Q';

execute 'gdxviewer.exe
i=C:\Users\Dell_\Documents\gamsdir\projdir\readData.gdx
csv=C:\Users\Dell_\Documents\gamsdir\projdir\resultsProd.csv id=X';

execute 'gdxviewer.exe
i=C:\Users\Dell_\Documents\gamsdir\projdir\readData.gdx
csv=C:\Users\Dell_\Documents\gamsdir\projdir\resultsInv.csv id=Inv';

execute 'gdxviewer.exe
i=C:\Users\Dell_\Documents\gamsdir\projdir\readData.gdx
csv=C:\Users\Dell_\Documents\gamsdir\projdir\resultsSub.csv id=S';
```

REFERENCES

Allahverdi, A., Gupta, J. N., & Aldowaisan, T. (1999). A review of scheduling research involving setup considerations. *Omega*, *27*(2), 219-239

Balas, E., Enos, L. T., & Graves, G. W. (1973, May). Computerized scheduling of printing operations. In Spring Conference of the Graphic Communications Computer Association. Toronto, USA.

Biskup D., Herrmann J. & Gupta J. N. D. (2008). Scheduling identical parallel machines to minimize total tardiness. *International Journal of Production Economics, 115*(1), 134-142.

Cakici, E., & Mason, S.J., (2007). Parallel machine scheduling subject to auxiliary resource constraints. *Production Planning & Control, 18*(3), 217–225.

Chen, K., & Ji, P. (2007). A mixed integer programming model for advanced planning and scheduling (APS). *European Journal of Operational Research 181*(1), 515–522.

Daniels, R. L., Hoopes, B. J., & Mazzola, J. B. (1996). Scheduling parallel manufacturing cells with resource flexibility. *Management Science, 42*(9), 1260–1276.

Darbi-Dowman, K. D., Little, J., Mitra, G., & Zaffalon, M. (1997). Constraint logic programming and integer programming approaches and their collaboration in solving an assignment scheduling problem. *Constraints, 1*(3), 245–264.

Davis, J., Edgar, T., Porter, J., Bernaden, J., & Sarli, M. (2012). Smart manufacturing, manufacturing intelligence and demand-dynamic performance. *Computers & Chemical Engineering*, 47, 145-156.

Deane, R. H., & White, E. R. (1975). Balancing workloads and minimizing setup costs in the parallel processing shop. *Operations Research Quarterly, 26*(1), 45-53.

Dearing, P. M., & Henderson, R. A. (1982). Assigning looms in a textile weaving operation with changeover limitations. Technical Report No 386, July, Department of Mathematical Sciences. Clemson University, USA 1982.

Dearing, P. M., & Henderson, R. A. (1984). Assigning looms in a textile weaving operation with changeover limitations. *Production and Inventory Management Journal, 25*(3), 23-31.

Doan, T. J. (2011). Risky Business: Evaluation of a Decision Support System for Use in a High Risk Environment (MA thesis). Retrieved from https://atrium2.lib.uoguelph.ca/xmlui/handle/10214/3000

Edis, E.B., & Oguz, C. (2012). Parallel machine scheduling with flexible resources. *Computers & Industrial Engineering, 63*(2), 433–447.

Edis, E.B., & Ozkarahan, I. (2011). A combined integer/constraint programming approach to a resource-constrained parallel machine scheduling problem with machine eligibility restrictions. *Engineering Optimization, 43*(2), 135-157.

Erdirik-Dogan, M., & Grossman, I. E. (2007). Planning Models for Parallel Batch Reactors with Sequence-Dependent Changeovers. *The American Institute of Chemical Engineers, 55*(9), 2284-2300.

Feng, G., & Lau, H. C. (2008). Efficient algorithms for machine scheduling problems with earliness and tardiness penalties. *Annals of Operations Research, 159*(1), 83-95.

Franca, P.M., Gendreau, M., Laport, G., & Muller, F. (1996). A tabu search heuristic for the multiprocessor scheduling problem with sequence dependent setup times. *International Journal of Production Economics, 43*(2-3), 79-89.

Galitz, W. O. (1985). *Handbook of screen format design*. Wellesley, MA: QED Information Sciences Inc.

Garey, M. R., & Johnson, D. S., (1979). Computers and Intractability: A guide to the theory of NP Completeness. San Francisco: W. H. Freeman.

Gedik, R., Rainwater, C., Nachtmann, H., & Pohl, E. A. (2016). Analysis of a parallel machine scheduling problem with sequence dependent setup times and job availability intervals. *European Journal of Operational Research, 251*(2), 640–650.

Gendreau, M., Laporte, G., & Guimara˜es, E.M. (2001). A divide and merge heuristic for the multiprocessor scheduling problem with sequence dependent setup times. *European Journal of Operational Research, 133*(1), 183–189.

Geoffrion, A. M., & Graves, G. W. (1976). Scheduling parallel production lines with changeover costs: Practical application of a quadratic assignment/LP approach. *Operational Research, 24*(4), 595-610.

Grigoriev, A., Sviridenko, M., & Uetz, M. (2005, June). Unrelated parallel machine scheduling with resource dependent processing times. *In International Conference on Integer Programming and Combinatorial Optimization* (pp. 182-195). Springer Berlin Heidelberg.

Guinet, A. (1984). Transport industriels routiers, un problem d'affection avec reemploys sous contraintes, *Revue d'Automatique d'Informatique et de Recherche Operationalle, 18*(4), 353-379.

Guinet A. (1990). Textile production systems: A succession of non-identical parallel processor shops. *Journal of Operational Research Society, 42*(8), 55-71.

Guinet, A., & Dussauchoy A. (1993). Scheduling sequence dependent jobs on identical parallel machines to minimize completion time criteria. *International Journal of Production Research, 31*(15), 79-94.

Hastings, N.A.J., Marshall, P., & Willis, R. J. (1982). Schedule Based M.R.P.: An Integrated Approach to Production Scheduling and Material Requirements Planning. *Journal of Operational Research, 33*(11), 1021-1029.

Heady, R. B., & Zhu, Z. (1998). Minimizing the sum of job earliness and tardiness in a multimachine System. *International Journal of Production Research, 36*(6). 1619-1632.

Hiraishi, K., Levner, E., & Vlach, M. (2002). Scheduling of parallel identical machines to maximize the weighted number of just-in-time jobs. *Computers & Operations Research 29*(7), 841-848.

Jain, V., & Grossmann, I.E., (2001). Algorithms for hybrid MILP/CP models for a class of optimization problems. *Informs Journal on Computing, 13*(4), 258–276.

Janakiraman, V. S., & Sarukesi, K. *Decision Support Systems*. Prentice Hall India Pvt., Limited, 2004. ISBN 9788120314443. Retrieved from http://books.google.ca/books?id=dx6E0Oe9yhIC

Johnson, S. M. (1954). Optimal two and three stage production schedules with setup times included. *Naval Research Logistics Quarterly, 1*(1), 61–8.

Josefowska, J., & Zimniak, A. (2008). Optimization tool for short-term production planning and scheduling. *International Journal of Production Economics. 112*(1), 109–120.

Kim, S. S., Shin, H. J., Eom, D. H., & Kim, C. O. (2003). A due date density-based categorizing heuristic for parallel machines scheduling. *International Journal of Advanced Manufacturing Technology, 22*(9-10), 753–760

Krajewski, L.J., King, B.E., Ritzman, L.P., & Wong, D. S. (1987). Kanban, MRP, and shaping the manufacturing environment. *Management Science, 33*(1), 39-57.

Kurz, M.E., & Askin, R.G. (2001). Heuristic scheduling of parallel machines with sequence-dependent set-up times. *International Journal of Production Research, 39*(16), 3747-3769.

Lara, A. F. B., Yalaoui, F., & Dugardin, F. (2012). A tabu search algorithm to minimize the total tardiness in the parallel machines scheduling problem. *IFAC Proceedings Volumes, 45*(6), 1371-1376.

Lasserre, J. B. (1992). An integrated model for job-shop scheduling planning and scheduling. *Management Science, 38*(8), 1201-1211

Lawler, E. L., Lenstra, J. K., Kan, A. H. R., & Shmoys, D. B. (1993). Sequencing and scheduling: Algorithms and complexity. *Handbooks in Operations Research and Management Science, 4,* 445-522.

Lawrynowicz, A. (2008). Integration of production planning and scheduling using an expert system and a genetic algorithm. *Journal of the Operational Research Society 59*(4), 455 – 463.

Lee, Y.H., Bhaskaran, K., & Pinedo, M. (1997). A Heuristic to Minimize the Total Weighted Tardiness with Sequence Dependent Setups. *IIE Transactions, 29*(1), 45-52.

Lee, Y.H., & Pinedo, M. (1997). Scheduling jobs on parallel machines with sequence-dependent setup times. *European Journal of Operational Research, 100*(3), 464-474

Lustig, I. J. & Puget, J.F., (2001). Program does not equal program: Constraint programming and its relationship to mathematical programming. *Interfaces, 31*(6), 29–53.

Marsh, J. D., & Montgomery, D. C. (1973). Optimal procedures for scheduling jobs with sequence-dependent changeover times on parallel processors. *AIIE Technical Papers, 1*, 279-286.

Mendes, A.S., Müller, F.M., França, P.M., & Moscato, P. (2002). Comparing meta-heuristic approaches for parallel machine scheduling problems. *Production Planning and Control, 13*(2), 143-154.

Omar, M. K., & Teo, S. C. (2007). Hierarchical production planning and scheduling in a multi-product, batch process environment. *International Journal of Production Research, 45*(5), 1029-1047.

Ovacık, I. M., & Uzsoy, R. (1994). Rolling horizon algorithms for a single machine dynamic scheduling problem with sequence dependent setup times. *International Journal of Production Research, 32*(6), 1243-1263.

Ovacik, I. M., & Uzsoy, R. (1995). Rolling horizon procedures for dynamic parallel machine scheduling with sequence-dependent setup times. *International Journal of Production Research 33*(11), 73-92.

Özdamar, L., Bozyel, M.A., & Birbil, S.İ, (1998). A hierarchical decision support system for production planning (with case study). *European Journal of Operational Research, 104*(3), 403-422.

Parker, R. G., Deane, R. H., & Holmes, R. A. (1977). On the use of a vehicle routing algorithm for the parallel processor problems with sequence dependent changeover costs. *AIIE Transactions 9*(2), 55-60.

Pol, A. & Ahuja, R. K. (2007). Principles of good database design. In *Developing web-enabled decision support systems* (pp. 3-12). Belmont, MA: Dynamic Ideas.

Power, D. J. (1998). Web-based decision support systems. *DSstar, The On-Line Executive Journal for Data-Intensive Decision Support, 2*(33).

Power, D.J. (2002). *Decision support systems: concepts and resources for managers*. Westport, CT: Quorum Books.

Radhakrishnan, S., & Ventura, J. A. (2000). Simulated annealing for parallel machine scheduling with earliness tardiness penalties and sequence-dependent set-up times. *International Journal of Production Research. 38*(10), 2233-2252.

Smith, B.M., Brailsford, S.C., Hubbard P.M., & Williams H.P. (1997). The progressive party problem: Integer linear programming and constraint programming compared. *Constraints, 1*(1), 119–138.

Sumichrast, R., & Baker, J. R. (1987). Scheduling parallel processors: an integer linear programming based heuristic for minimizing setup time. *International Journal of Production Research. 25*(7), 61-71.

Şeref, M. M. H., Ahuja, R. K. & Winston, W. L. (2007). The DSS development process. In *Developing spreadsheet-based decision support systems* (pp.571-594). Belmont, MA: Dynamic Ideas.

Tan, W. (2000). Integration of process planning and scheduling - a review. *Journal of Intelligent Manufacturing, 11*(1), 51-63

Turban, E., Aronson, J. (2001). *Decision Support Systems and Intelligent Systems*. Upper Saddle River, NJ, USA: Prentice-Hall.

Venkataraman, R. & Smith, S.B. (1996). Disaggregation to a rolling horizon master production schedule with minimum batch-size production restrictions. *International Journal of Production Research, 34*(6), 1517–1537.

Vignier, A., Sonntag, B., & Portmann, M. C. (1999). A hybrid method for a parallel-machine scheduling problem. In *Emerging Technologies and Factory Automation, 1999. Proceedings. ETFA'99. 1999 7th IEEE International Conference on* (Vol. 1, pp. 671-678). IEEE.

Watters, L.J. (1967). Reduction of integer polynomial programming problems to zero-one linear programming problems. *Operations Research 15*, 1171–1174.

Wilbrecht, J. K., & Prescott, W. B. (1969). The influence of setup time on job shop performance. *Management Science, 16*(4), 274-280.

Xue, G., Offodile, O.F., Zhou, H., & Troutt, M.D. (2011). Integrated production planning with sequence-dependent family setup times. *International Journal of Production Economics*, *131*(2), 674–681

Yalaoui, F. (2012). Minimizing Total Tardiness in Parallel-Machine Scheduling with Release Dates. *Applied Evolutionary Computation, 3*(1), 21-46.

Yang, Y., Parsaei, H., & Leep, H. (2001). A prototype of a feature-based multiple-alternative process planning system with scheduling verification. *Computers and Industrial Engineering, 39*(1-2), 109–124.